STEREOSCOPIC VIDEO CODING

By

Roland Siu-Kwong IP

A Thesis

submitted in partial fulfillment of the requirements

for the Degree of Master of Philosophy

Division of Electronic Engineering

The Chinese University of Hong Kong

October 1995

Acknowledgement

I would like to thank my project supervisor, Dr. S. M. Chiang, for his guidance, suggestions, advice and constructive criticism throughout the course of this research.

I am grateful to parents, my brothers for their spiritual support and encouragement.

Last but not the least, I would like to thank Yee Man, my wife, without whose love, encouragement and understanding, this study could not have been done.

Abstract

Stereo imagery consists of a moving image sequence for the right eye and another moving image sequence for the left eye forming a moving 3-D view. The MPEG video compression standard defines the coding of a single sequence of moving images and associated audio for digital media at up to 1.5Mbps. If a pair of stereo image sequences are directly encoded separately to MPEG format, there will be two MPEG bitstreams in result requiring twice the bandwidth for transmission or twice the capacity for storage. This thesis describes a new video coding scheme for stereo moving image sequences providing better coding efficiency and with the resultant bitstream compatible to existing MPEG decoders for monocular viewing.

The coding scheme makes use of disparity compensation technique to eliminate the similarities between a pair of left and right stereo moving image sequences. By using left image sequence as reference, the disparities of the image sequence pair are computed so that together with the reference image sequence, the right image sequence can be reproduced. Thus, the original right image sequence is redundant and can be discarded for storage or transmission. The reference left image sequence is then encoded in MPEG format with the disparities inserted at fields reserved for user data forming an MPEG compatible stereoscopic video bitstream.

At the decoder, the disparities is extracted and combined with the reference image sequence to generate the disparity compensated right image sequence forming a pair of stereo moving image sequences. Since the disparities are encoded as user data, when the

stereo video bitstream is being displayed using a monocular MPEG decoder, the user data is simply discarded and the reference left image sequence will be shown. Both objective and subjective performance evaluations for the coding scheme were conducted by constructing a software prototype encoder. The objective evaluation was performed by comparing the peak signal-to-noise ratio while the subjective evaluation was performed by displaying the decoded left and right image sequences on the screen and visually comparing their image qualities. The result showed that for a pair of stereo image sequences, the new stereo coding scheme could achieve a 15% further compression, comparing to coding the two image sequences independently in MPEG format, with no substantial degrade in image quality.

Contents

Chapter 1

Introduction

1.1 Motivation

With the increasing use of digital video in multimedia systems, there is a growing need for efficient coding methods. The rising application of virtual reality requires even two channels of motion video to provide the user a three-dimensional vision in a computer generated environment. Direct digitized motion video requires a large amount of storage space and transmission bandwidth, therefore special coding scheme is necessary to reduce size of the digital video data. Digital video standards such as CCITT H.261 and MPEG are two widely accepted schemes developed for this purpose.

Stereoscopic video system provides two channels of video sequence, one for each eye, to give the viewer extra sense of depth perception. When these two channels of video sequence are coded independently, the transmission and/or storage require twice the bandwidth and/or storage space. As the two channels of video sequence are moving views with slightly displaced view points, they contain many similarities. A more compact stereoscopic video bitstream can be generated by removing these similarities.

In view of the lack of standard coding scheme for stereoscopic video coding, which is essential for the utilization of stereoscopic motion video and having observed that the

MPEG standard has taken a firm root in digital video applications, designing a stereoscopic video format compatible to MPEG has an advantage that existing MPEG video systems can easily be upgraded to stereo capability for stereo applications without loosing the compatibility with existing monoscopic MPEG applications.

1.2 Image Compression

The generation of motion video can be considered as a set of image frames displayed sequentially at a rate not less than 25 frames per second. Thus, the techniques for still image compression can also be applied to each of the frames in the sequence for compression. Still image compression studies how to map original digital image to the coded representation so that the number of bits required is minimized. The following subsection describes some popular ways to achieve the compression.

1.2.1 Classification of Image Compression

Basically, image compression can be divided into two types: *Lossless Compression* and *Lossy Compression*.

- **Lossless Compression** (also known as entropy coding or invertible coding), this is where the original image can be perfectly recovered from its coded representation. Factors of human perception, therefore, play no role in developing this type of compression schemes. Coding techniques such as *Huffman coding* [1], *run-length coding*, *arithmetic coding* and *Ziv-Lempel coding* [2] belong to this type. The principle of these techniques is to use long code words to represent less likely inputs and short code words to represent more likely inputs. They are ideal for applications, such as in medical imaging and in scientific applications. However, the typical compression ratio achieved by them is moderate, which is between 1.7 to 2.1. The ultimate limits to this type of compression are determined by the Shannon's principle [3], which says that loss is inevitable if the transmission bit rate is smaller than the entropy of the source.

- **Lossy Compression**, this is where the original image frame cannot be perfectly recovered from the coded representation. The image restored from the compression process is only an approximation to the original according to some fidelity criterion. Depending on the quality required, achievable compression ratio ranges from 2 to 100. Most existing image coding schemes utilize this type of compression.

Unlike applications in medical imaging and in scientific applications, which require severe analyses of the image data, the final destination of general image system as well as video system is human eyes. Because of the deficiencies of human visual perception, lossy compression is acceptable. In the following discussion, lossy compression will be explored further.

1.2.2 Lossy Compression Approaches

In addition to the entropy coding, lossy compression technique reduces the redundancy within an image frame. The sources of redundancy include spatial and color space. Spatial redundancy comes from the fact that nearby image data are correlated with each other. While color space redundancy describes that the RGB components of pixels in an image frame are correlated among themselves. Color space redundancy can be reduced by using other color coordinate systems to represent the RGB components such that bandwidths for their transmission are smaller than those for the original RGB components. Y-Cr-Cb system, with Y representing the luminance, Cr and Cb representing the chrominances, is an example of such a system. Spatial redundancy reduction can be directly performed on a two-dimensional, discrete distribution of image data which are the sampling representation of the 2D signal waveform of still image frame. Image coding methods based on this strategy are known as *Waveform Based Coding*. Most image coding techniques, such as predictive coding, transform coding and subband coding belong to this type.

1.3 Video Compression

A moving image is formed by a set of still image displayed in sequence. We may compress the image sequence one by one using one of the image compression methods to achieve overall compression. This is known as intraframe processing. However, the efficiency so yielded is far from practical. Since a video sequence or moving image sequence is captured from a scene at a sequence of time intervals, it contains not only spatial and color space redundancies, but also temporal redundancy. To exploit this redundancy, image model can be employed and model parameters can then be extracted for representation. Based on different models, redundancy existing in an image sequence can be reduced at different levels. For instance, an image sequence can be represented based on a 3D real physical scene model. For the transmission of a "head-and-shoulder" image found in videophone applications, a 3D model of the object can be built and parameters representing the texture, and the position as well as orientation of edges can be transmitted. At the decoder, each frame of the image sequence is synthesized with the received parameters. Thus, much temporal redundancy contained in the image sequence can be removed. This coding strategy is known as *Model Based Coding*.

Waveform based coding which can be used to reduce spatial redundancy can also be applied to remove temporal redundancy. In this case, a segment of an image sequence is considered as a 3D signal waveform including the 2D signal waveform to represent each each image frame and the additional dimension for the temporal domain. Take 3D transform coding as an example. The 3D signal waveform can be directly transformed to 3D transform coefficients, which having more compact distribution of energy comparing to the original image data. More significant coefficients are retained while the less significant coefficients can be discarded. Other temporal redundancy reduction technique, such as motion compensation operation, can also be incorporated with waveform based coding. This section only gives a brief introduction to the approaches. More detail waveform based coding and model based coding techniques will be described in the next chapter.

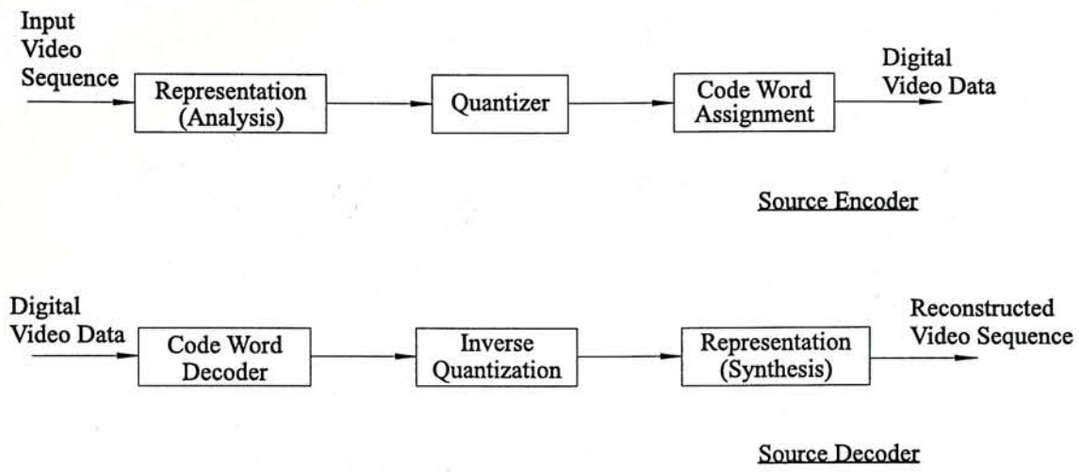


Figure 1.1: *General Video Compression System*

1.3.1 Video Compression System

Both waveform based coding and model based coding describes ways to represent video information. Two more processes: quantization and code word assignment are usually required. These three distinct processes are the main elements of any digital video compression system. Figure 1.1 depicts the general digital video compression system. Input image sequence is the digitized version of video. Through the representation (analysis) process, it is expressed in a more efficient representation which can be the model parameters for model based coding or the transform coefficients for the transform coding. The resultant data with the less important data discarded are then fed into the quantization process. This process performs the discretization of the transmitted data so that a smaller set of data results. This can be performed one parameter at a time by *scalar quantization*, or a group of parameters at a time by *vector quantization*. The last process assigns an appropriate code word for each of the quantized data. Entropy coding is usually applied to reduce the average bits per symbol. At the decoder, image sequence is reconstructed by performing the inverse processes.

Temporal processing plays an important role in video compression. It is one of the elements in the representation synthesis function block shown in Figure 1.1. For high efficient video coding, temporal processing is essential. Although intraframe processing

only video coding has a number of benefits such as avoiding the complexity of temporal processing and getting rid of the need for extra frame stores at the encoder and decoder, only the spatial domain redundancy can be exploited and the temporal domain redundancy is ignored. High-performance video compression standards, such as MPEG, utilize temporal processing to achieve high compression ratio.

1.4 Stereoscopic Video Compression

Stereoscopic video consists of two channels of motion video. It can be seen that its compression can be done by treating the two channels of video as two separate sequences of moving images coded independently. However, this method only exploits the spatial and temporal redundancies; the correlation between the two channels of motion video is ignored. The source of the correlation is due to the fact that the two channels of motion video are views of a scene at a sequence of moments with slightly displaced view points. By utilizing this correlation, more efficient compression algorithm can be built.

Model based coding for single channel video compression can be used to exploit this correlation by adding more parameters such as the angle of the views and positions of the cameras to describe the view points of the cameras. At the decoder, each frame of the two image sequences is synthesized from the model and the parameters received. By treating the image frames in the two channels of image sequences as 2D signal waveforms, waveform based coding can also be applied by predicting the image frames in one of the two channels from the other. This will be explained further in chapter 3.

1.5 Organization of the thesis

Following this introductory chapter, chapter two describes the basis of video coding theories that used throughout this study. It begins with representation analysis for video compression in which temporal and spatial redundancy reductions are investigated

and followed by the implementation of quantization. Then possible ways for code word assignment is described. Finally, the selection of video coding standard for compatibility is detailed.

In chapter three, attention is concentrated on the development of an MPEG compatible stereoscopic coder. The stereoscopic video coding scheme is presented, in which an algorithm to estimate disparity is proposed. The corresponding decoding algorithms based on the existing MPEG standard are then given.

In chapter four, objective and subjective results are compared and analyzed. The effect of varying the bit rates of the additional channel to the decompression quality is discussed.

The thesis concludes with chapter five which collates the discoveries and work that has been performed in the course of the research program, and makes suggestions for further improvement.

Chapter 2

Motion Video Coding Theory

2.1 Introduction

This chapter explains the basic theories of the existing monocular or single channel motion video coding schemes and how they take advantage of short-falls in the human visual system. In the previous chapter, it is mentioned that the general framework for designing any digital video coding algorithm includes three main elements: representation of the image sequence, quantization and code word assignment. The following section elaborates on efficient representations of motion video and their applicability for processing along the temporal and spatial dimensions is investigated. Section 2.3 examines the quantization of the parameters of the representation. Then, code word assignment of the quantized parameters is discussed in section 2.4. Finally, selection of video coding standard for implementation is given in section 2.5.

2.2 Representations

The aim of designing a representation for motion video compression application is to minimize the redundancy by distributing the maximum amount of perceptually important contents into a small fraction of the parameters. The more significant parameters

may be retained for further processing while the less significant ones may be simply discarded. There are a number of ways to implement representation. They depend on the specific application and the implementation constraints. In the previous chapter, the representation of moving image sequence is divided into two main categories: model based coding and waveform based coding. Waveform based coding can further be subdivided into predictive coding and transform/subband coding. Among them the most complex and sophisticated scheme is model based coding, followed by transform/subband coding. Predictive coding is the simplest and easiest one to be implemented. In this section, the various approaches for developing the representation process is discussed. Furthermore, the identification of the important information to be transmitted is given. For each approach, the possible application along temporal and spatial dimensions is investigated.

Predictive Coding

Generally speaking, motion video can be considered as stationary in its characteristics over a small region in an image frame (spatial dimension) and a short period of time (temporal dimension). Predictive coding exploits these stationary features. For spatial operation, predictive coding considers a complete image frame line by line as a continuous sequence of pixels. Each current pixel value such as intensity value is predicted from the previous pixel values. The number of previous pixel values required depends on the implementation of the predictor. While for temporal operation, predictive coding predicts the current image frame from the previous image frame. The differences between the predicted pixels (or image frame) and the actual corresponding pixels (or image frame) are the new information, which is called *residual*. This residual is coded and transmitted to update the prediction. Figure 2.1 shows the block diagram of a typical predictive coding system. The main idea behind all such systems is to form a prediction and then encoding the residual. Obviously, the coding efficiency is primarily determined by the accuracy of the predictor. Moreover, due to the recursive nature of predictive coding, the decoder has to track the encoder accurately; otherwise error will accumulate.

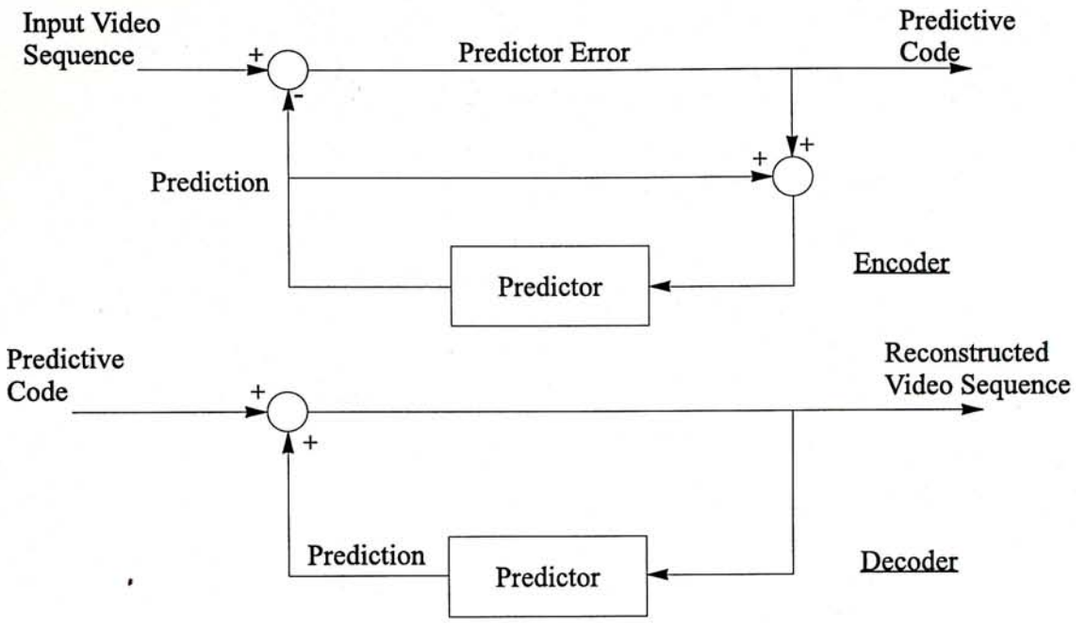


Figure 2.1: *Predictive Coding System*

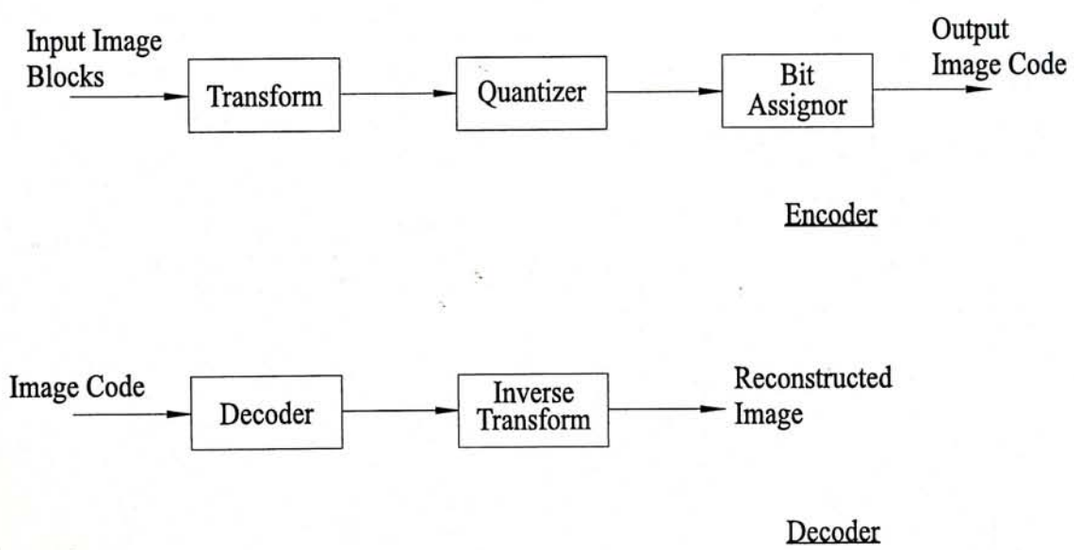


Figure 2.2: *Transform Coding System*

Transform/subband Coding

In transform/subband coding, a completely different approach is used. Figure 2.2 shows the block diagram of a transform coding system. It is based on modifying the transform of an image. A reversible linear transform such as the Karhunen-Loève (KLT), Discrete Fourier (DFT), and Discrete Cosine (DCT) is used to map the images into a set of transform coefficients, which are then quantized and coded. Since most of the energy (and information) is concentrated in a small fraction of the transform coefficients, high-quality images may be reconstructed with minimal distortion from few energetic coefficients. The spatial characteristics are exploited by dividing an image frame into subpictures of a particular block size, 8×8 or 16×16, which is independently transformed and adaptively processed.

Subband coding was first introduced to image coding by Vetterli [4]; Wood and O'Neill [5] in 1986. Using a number of filters, the process splits the incoming image into separate frequency bands or subbands. It can be seen that the output coefficients of the transform coding system and the set of channel outputs of the subband coding are the decomposition of images into its frequency and subband components, respectively, with energy redistribution. Thus, they are classified to be the same type. A one-dimensional four band decomposition and reconstruction subband coding scheme is shown in Figure 2.3. When the signal is divided into subbands, the outputs are encoded adaptively to exploit the specific characteristics of each subband.

Model Based Coding

The most complex coding method, model based coding depends on an algorithm that decomposes a video into features, such as contours, textures, human faces, and other 3D scene model. Then, parameter encoder is employed to encode those features. Only few important parameters are transmitted under some quality criteria. At the decoder, the received parameters are decoded and synthesized to a reconstructed video by applying the same model. Figure 2.4 shows the basic schematic diagram. The efficiency of this

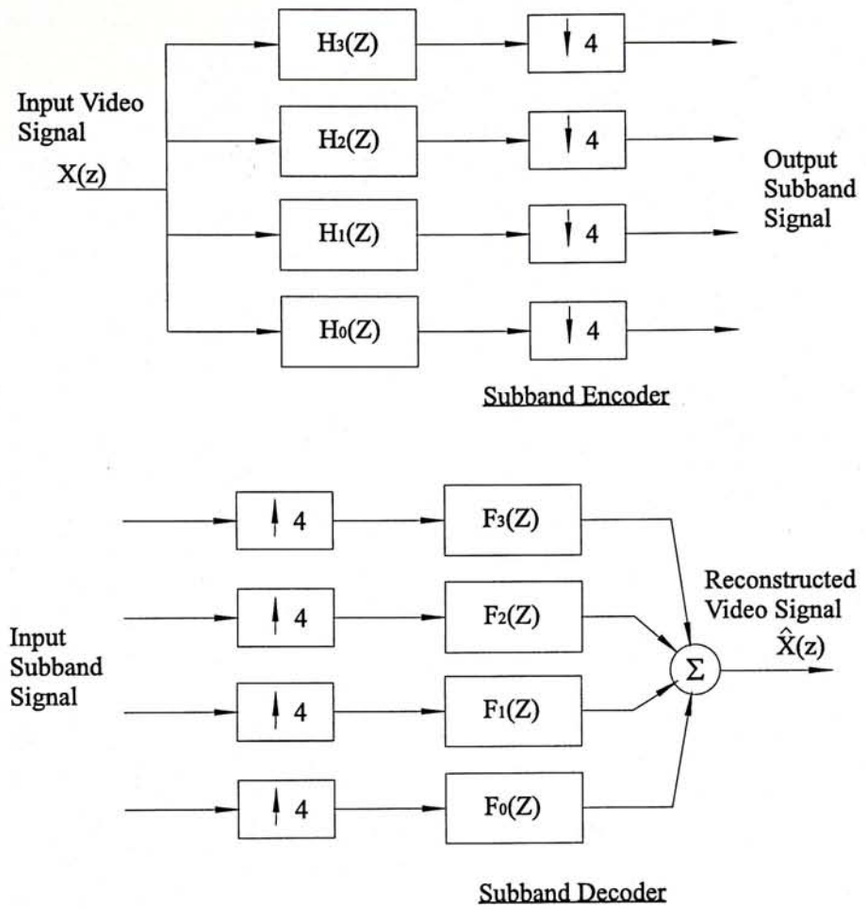


Figure 2.3: *Subband Coding System*

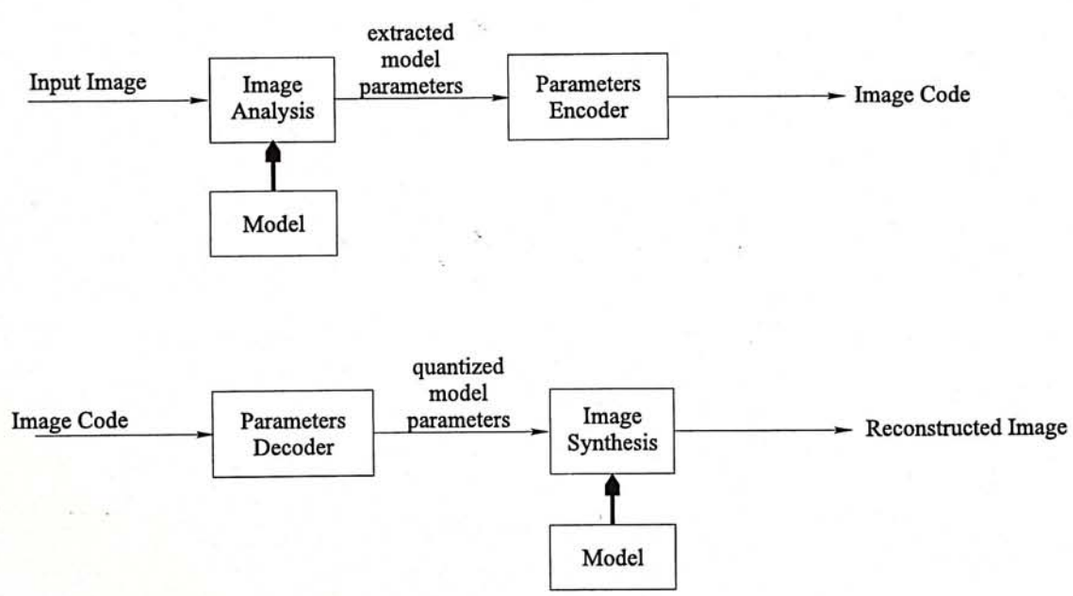


Figure 2.4: *Model Based Coding System*

type of coding depends on the accuracy of the model to the video being coded. It is believed that model based coding is able to achieve substantially greater compression than the previously discussed coding schemes if an accurate model can be found. For situations where the model does not match with the video sequence, very poor quality video will result or high bit rate/storage space is required.

2.2.1 Temporal Processing

After intraframe processing, temporal processing is necessary to reduce the temporal domain redundancies. It is achieved by discarding the the stationary regions of successive frames. One way, for example, is to compute the differences between image frames and code them accordingly. There are more efficient ways to reduce temporal redundancy and some of them use the coding methods described in the previous sections. The following discussion depicts the use of the spatial coding methods in the temporal domain.

Predictive Coding

By using predictive coding, temporal redundancy is reduced by motion compensation. Motion compensation is the processing of individual frames while compensating for the presence of motion. It requires a process of estimating the motion which is known as motion estimation. Motion estimation is based on the assumption that successive video frames contain the same image contents at the same positions or slightly shifted position across the frame. An image frame is possible to be predicted from previous frame by estimating the motion for each small region within the frame. There are two algorithms: Pixel Recursive Algorithms (PRA) and Block Matching Algorithms (BMA). PRA deals with the motion of individual pixel, where the motion of a pixel in a frame is an update of earlier motion information on the same or neighboring pixels. Due to the intensive computation involved, PRA is seldom used in practice. On the other hand, BMA is more widely used, and in particular, both MPEG and H.261 employ this algorithm for motion estimation. BMA partitions an image frame into blocks of $M \times N$ pixels and considers the motion of each block relative to the preceding image frame. The actual

process is done by finding the "best match" and displacement (or motion vector) for each block in an image frame from the preceding frame. Cost function such as minimum mean squared error (MSE), minimum mean absolute error (MAE), or maximum cross correlation function (CCF) is used in the algorithm to find the "best match". They are defined as follows:

Let pixel block size be $M \times N$ pixels.

Mean absolute error (MAE),

$$M_1(i,j) = \frac{1}{MN} \sum_{m=1}^{M} \sum_{n=1}^{N} |U_C(m,n) - U_R(m+i,n+j)|, \qquad -p \le i,j \le p. \tag{2.1}$$

Mean squared error (MSE),

$$M_2(i,j) = \frac{\sum_{m=1}^{M} \sum_{n=1}^{N} [U_C(m,n) - U_R(m+i,n+j)]^2}{\sum_{m=1}^{M} \sum_{n=1}^{N} U_C^2(m,n)}, \qquad -p \le i,j \le p. \tag{2.2}$$

Cross correlation function (CCF),

$$M_3(i,j) = \frac{\sum_{m=1}^{M} \sum_{n=1}^{N} U_C(m,n) U_R(m+i,n+j)}{\left[\sum_{m=1}^{M} \sum_{n=1}^{N} U_C^2(m,n)\right]^{1/2} \left[\sum_{m=1}^{M} \sum_{n=1}^{N} U_R^2(m+i,n+j)\right]^{1/2}}, \qquad -p \le i,j \le p, \tag{2.3}$$

where $U_C(m,n)$ and $U_R(m,n)$ are unit pixel quantity measures, such as RGB values or YCrCb values, at row m and column n of the current frame and the reference frame, respectively. The basic block matching geometry is illustrated in Figure 2.5. Each block of size $M \times N$ is compared with all $M \times N$ segments of the previous frame that lie within a search area of size $(M+2p) \times (N+2p)$. The simplest and straight forward way to find the "best match" is called *full search*, which examines every possible candidate within the search area. Koga *et al.* [6] proposed a faster algorithm called *three-step search*, which is illustrated in Figure 2.6. In the first step the 9 shift values labeled s1 are considered (4 corner points, 4 middle points of the sides, and the center sample). In the second step the size of the square is reduced by 1 and the center point of the square

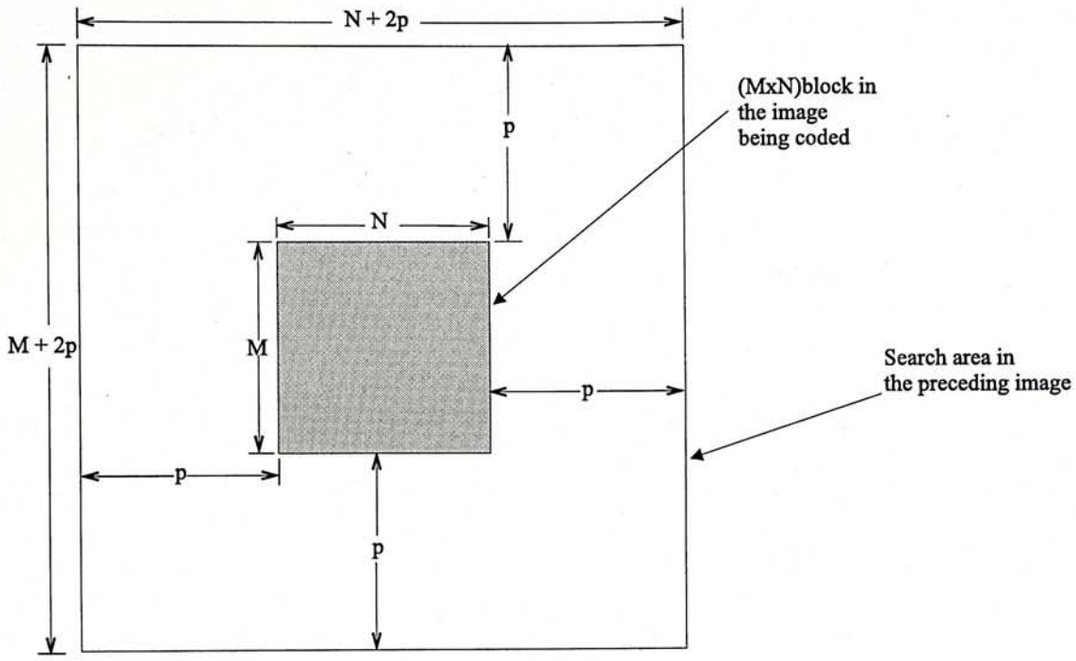


Figure 2.5: *Block matching geometry – coding block and the search area.*

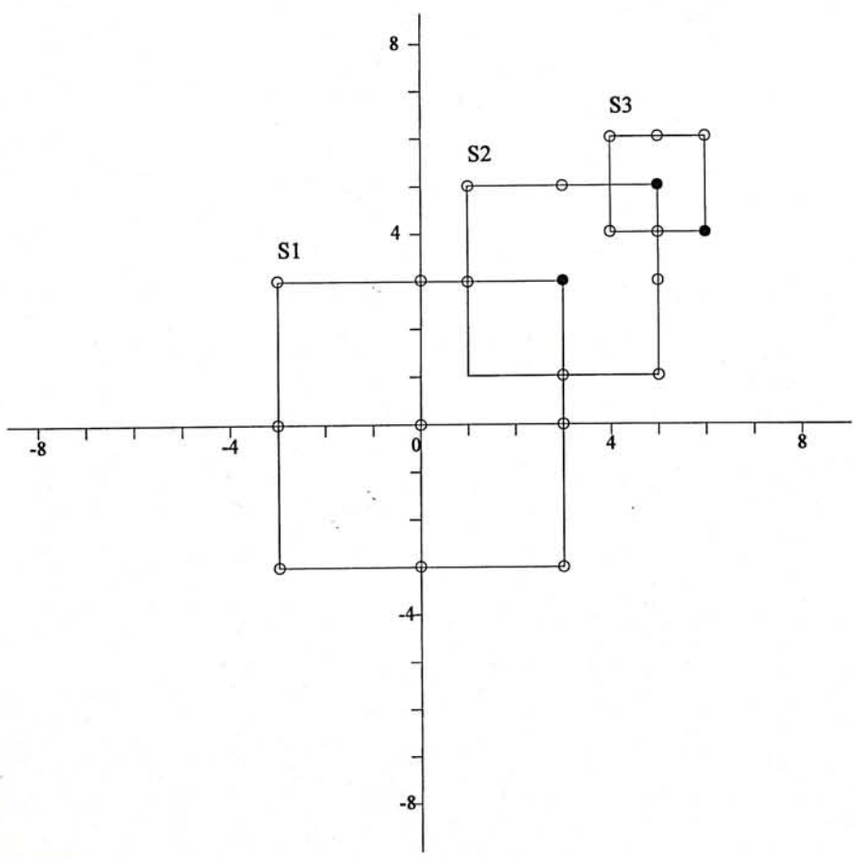


Figure 2.6: *The three-step search algorithm.*

is moved to the best matching point of the previous stamp and the 8 new points are searched. A similar procedure is followed in the third step. Jain and Jain [7] proposed *the logarithmic search*, which tracks the direction of minimum distortion. It reduces the computational requirements considerably. Srinivasan and Rao [8] proposed an algorithm called *the conjugate direction search*, which determines the minimum distortion locations along the conjugate directions successively.

After motion estimation, motion compensation is applied to predict the current frame by compensating for the motion in some areas among successive frames of video sequence based upon the motion vectors and the blocks in the reference frame(s) which can be a preceding frame only for causal prediction (*or forward prediction*) or a preceding frame as well as a succeeding frame for bidirectional prediction. Figure 2.7 shows the operation of causal prediction, where the predicted current frame consists of displaced blocks from the preceding frame. Ideally, it is hope that a match can be found for each pixel block to a preceding reference image frame so that a reconstruction can be made to create the present frame with the motion vectors alone. This will give maximum compression. However, it is seldom the case in practice. It is more likely that good matches cannot be found for some pixel blocks and prediction error (*or residual*) is usually required to be transmitted or stored for the correction.

Figure 2.8 shows the operation of bidirectional prediction, where the block of current frame is the average of the displaced block from the preceding frame and the displaced block from the succeeding frame. Since some of the occluded area which cannot be predicted from the preceding frame are predictable from the succeeding frame, bidirectional prediction usually generates a predicted frame with less error. This error is small enough to be ignored at all. In the case of a complete frame being discarded, bidirectional prediction is functionally similar to an interpolation process. Although interpolation can give a significant amount of compression by discarding frames, it reduces the temporal correlation between the remaining frames; accordingly coding for them are more difficult.

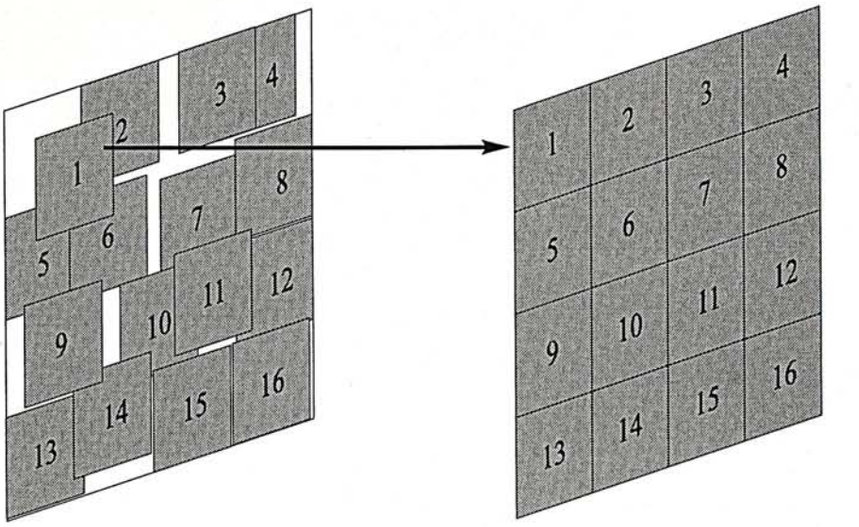


Figure 2.7: *Causal MC-prediction.*

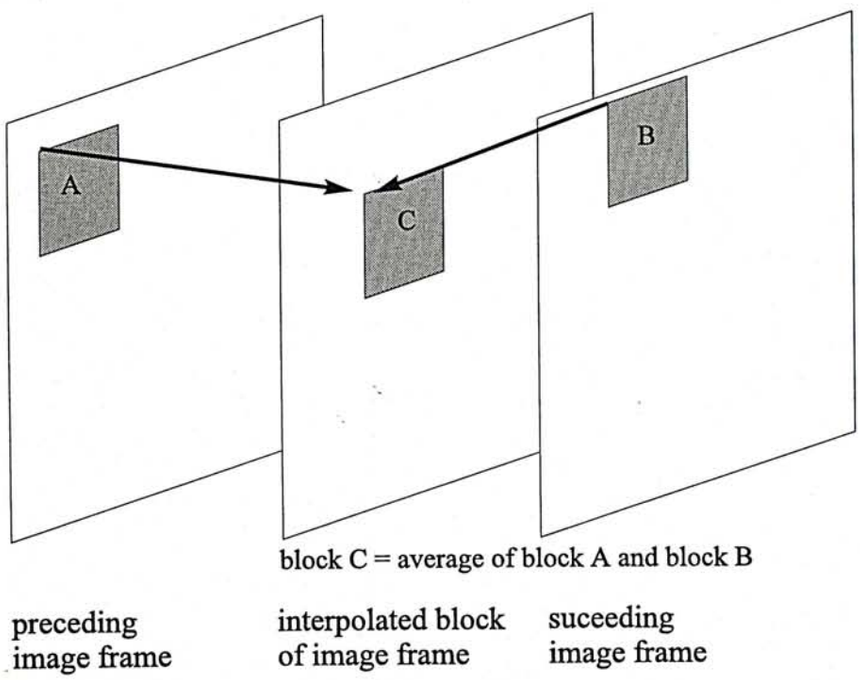


Figure 2.8: *Interpolated block constructed by MC-interpolation.*

Other drawbacks include more computational requirement and extra frame memory comparing to the forward processing of causal prediction. In spite of the efficiency of motion compensation in removing temporal redundancy, it cannot be applied at a scene change. In this case the original frame should be coded instead. Criteria have to be defined to identify which frame can be applied instead of discarding the frames periodically. Recently, Olstad [9] proposed to adaptively discard frames depending on the local activity, which is a measure of homogeneity of image sequence of consecutive intensity. Since the proposed method processes a large number of frames at a time, its implementation requires even more computing memory.

Transform/Subband Coding

Clearly, the temporal dimension can be considered as an additional dimension in a transform coding system so that three-dimensional transform coding results. Actual image sequences were simulated via this process by Roese *et al.* [10] and Natarajan *et al.* [11], but due to the computational complexity and extensive computing memory required to process and store the frames for temporal domain, it is seldom used in practice. In subband coding system, the temporal domain can also be applied to the subband filters so that the frequency components depend on column, row and frame. This process suffers the same limitations as for transform coding system discussed above. As only two frame stores are required for motion prediction, it is usually combined with transform/subband coding system to reduce temporal redundancy.

Model Based Coding

Temporal redundancy reduction for model based coding system requires the estimation of three-dimensional motion of the objects in the scene. Two approaches were proposed for this purpose: based on special features [12] and based on the displacement vector field (*optical flow*) [13]. In the first approach, special features such as corresponding points and lines are extracted from each image frame for the computation of the motion parameters. The second approach based on computing the optical flow, which is then

used in conjunction with additional constraints to compute the actual three-dimensional relative velocities between objects in the scene. The computation of the optical flow involves evaluating first and second partial derivatives of image brightness values which is time-consuming. Thoma and Bierling [14] suggested to compute the displacement vector field based on hierarchical block matching; this is similar to motion estimation discussed for predictive coding system.

2.2.2 Spatial Processing

Spatial processing is a process to reduce spatial redundancy, which exists in the residual after temporal processing. Predictive method for still image as described earlier may be used to reduce the spatial redundancy. However, they cannot give a satisfactory performance due to the fact that the error in predicting each residual sample requires a minimum of one bit to be represented; they do not operate at fractional bit. Transform/subband coding gives better results.

Residual contains strong spatial correlation like an image frame. It is more efficient to apply the transform/subband coding techniques. The following subsection describes the application of the transform and subband coding techniques for spatial processing.

Transform Domain Coding Scheme

Transformation is the most important part in the transform coding scheme. The main purpose of transformation is to convert statistically dependent information contents of an image frame into an array of uncorrelated coefficients so that maximum energy is packed into a minimum number of coefficients. The Fourier transform is possibly the most powerful tool in signal analysis. For digital video sources, the digitized two-dimensional version or the two-dimensional Discrete Fourier transform (DFT) has to be used. For an $N \times N$ image $u(m,n)$, the two-dimensional DFT is a separable transform defined as

$$v(k,l) = \sum_{m=0}^{N-1} \sum_{n=0}^{N-1} u(m,n) W_N^{km} W_N^{ln}, \qquad 0 \leq k,l \leq N-1. \tag{2.4}$$

where

$$W_N \triangleq \exp\left\{\frac{-j2\pi}{N}\right\} \tag{2.5}$$

and the inverse transform is

$$u(m,n) = \frac{1}{N^2}\sum_{k=0}^{N-1}\sum_{l=0}^{N-1} v(k,l)W_N^{-km}W_N^{-ln}, \qquad 0 \leq m,n \leq N-1. \tag{2.6}$$

There are two disadvantages concerning the use of 2D-DCT. Firstly, complex coefficients are involved. Secondly, Lim [15] showed that DFT is inherent inefficient in its energy compaction.

Among all linear transforms, the optimum in terms of energy compaction and decorrelation of the transform coefficients is the Karhunen-Loève transform (KLT), which was introduced by Karhunen [16] and Loève [17]. Hotelling [18] suggested the discrete equivalent of the KLT; therefore, discrete KLT is also called *Hotelling transform*. Considering an $N \times N$ image as N random vectors of the form

$$\boldsymbol{x} = \begin{bmatrix} x_1 \\ x_2 \\ \vdots \\ x_N \end{bmatrix} \tag{2.7}$$

and the mean vector $\boldsymbol{m}$ of the N vectors is defined as

$$\boldsymbol{m} = E\{\boldsymbol{x}\} \tag{2.8}$$

where $E\{arg\}$ is the expected value (or *mean*) of the argument. Thus, $\boldsymbol{m}$ can be expressed as

$$\boldsymbol{m} = \frac{1}{N}\sum_{k=1}^{N} \boldsymbol{x}_k \tag{2.9}$$

where $\boldsymbol{x}_k$ denotes the kth vector.

The *covariance matrix* $\boldsymbol{C}$ of the vectors is defined as

$$\boldsymbol{C} = E\left\{(\boldsymbol{x}-\boldsymbol{m})(\boldsymbol{x}-\boldsymbol{m})^T\right\} \tag{2.10}$$

where T indicates vector transposition. As $\boldsymbol{x}$ is an N dimension vector, $\boldsymbol{C}$ as well as $\left\{(\boldsymbol{x}-\boldsymbol{m})(\boldsymbol{x}-\boldsymbol{m})^T\right\}$ are matrices of order $N \times N$. The (i,j)th element of $\boldsymbol{C}$ is the covariance between elements x_i and x_j and the (i,i)th element of $\boldsymbol{C}$ is the variance of x_i, the ith component of the N vectors. The covariance matrix can also be expressed as follows:

$$\boldsymbol{C} = \frac{1}{N}\sum_{k=1}^{N}\left\{\boldsymbol{x}_k\boldsymbol{x}_k^T - \boldsymbol{m}\boldsymbol{m}^T\right\} \tag{2.11}$$

A transform matrix $\boldsymbol{A}$ whose rows are formed from the eigenvectors of $\boldsymbol{C}$ is an optimal transform, which results in completely uncorrelated coefficients. Let $\boldsymbol{y}$ be the uncorrelated transform coefficient vectors. The following equation using the transform matrix $\boldsymbol{A}$ is the Hotelling Transform equation.

$$\boldsymbol{y} = \boldsymbol{A}(\boldsymbol{x}-\boldsymbol{m}) \tag{2.12}$$

where $\boldsymbol{x}$ and $\boldsymbol{m}$ are the vectors defined in Equations (2.7) and (2.8), respectively. Because of the real and symmetric nature of the matrix $\boldsymbol{C}$, it is possible to find a set of N orthonormal eigenvectors analytically [19]. However, there is no known fast algorithm to perform transformation and the eigenvectors depend on the input image data; therefore, it is required to compute the eigenvectors for different data set. The computation requires extra computational time. All the above problems prevent the Hotelling transform from being used in practice. Instead, suboptimum transformations, which are not perfect decorrelators, are usually utilized.

The Discrete Cosine Transform (DCT) unlike the Hotelling transform has predetermined eigenvectors (or basis images). Ahmed *et al.* [20] pointed out that the KLT basis images of a first-order Markov image source is very similar to the DCT basis images. Clarke [21] further showed that as the correlation between adjacent pixels approaches one, the input dependent KLT basis images become identical to the input independent DCT basis images. Hence most practical transform coding systems including CCITT H.261 and MPEG are based on the DCT.

The two-dimensional DCT for an $N \times N$ image $u(m,n)$ is expressed as follows:

$$v(k,l) = \alpha(k)\alpha(l) \sum_{m=0}^{N-1} \sum_{n=0}^{N-1} u(m,n) \cos\left[\frac{(2m+1)k\pi}{2N}\right] \cos\left[\frac{(2n+1)l\pi}{2N}\right], \quad 0 \le k,l \le N-1 \tag{2.13}$$

and the inverse transform is

$$u(m,n) = \sum_{k=0}^{N-1} \sum_{l=0}^{N-1} \alpha(k)\alpha(l)v(k,l) \cos\left[\frac{(2m+1)k\pi}{2N}\right] \cos\left[\frac{(2n+1)l\pi}{2N}\right], \quad 0 \le m,n \le N-1. \tag{2.14}$$

In both equations (2.13) and (2.14), α is

$$\alpha(k) = \begin{cases} \sqrt{\frac{1}{N}} & \text{for } k = 0 \\ \sqrt{\frac{2}{N}} & \text{for } k = 1,2,\ldots,N-1. \end{cases} \tag{2.15}$$

The DCT has the advantages of having fast algorithm and good energy compaction properties. It is typically calculated over subblocks of an image frame. Since if the entire frame is transformed, the characteristics within the whole frame is considered equally. Those characteristics usually vary with the location of regions although the characteristics within a region is regarded as stationary. To exploit this nature, the DCT is applied to a smaller blocks usually with sizes 8×8 or 16×16 pixels. Performing transform in this way has other benefits, including reduced computational time and memory requirement. The side effect of the block DCT is the blocking artifact, that results when an image frame is segmented into numerous smaller blocks, each of which is independently transformed, the boundaries between those blocks become visible after inverse transform. Malvar and Staelin [22] suggested to reduce the blocking effect by Lapped Orthogonal Transforms, which is based on overlapping the basis functions to those of adjacent blocks. Subband coding can also be applied to reduce this type of artifact.

Subband Filtering Coding Scheme

As shown earlier in Figure 2.3, the analysis stage of the subband coding scheme computes subband transforms by filtering the input images with a set of bandpass filters, followed

by down-sampling (*or decimating*) the results. The transfer function of the filters correspond to the basis functions of the transform coding scheme. Each of the subband images represents a particular portion of the frequency spectrum. Owing to the capability to code each subband separately with a bit rate that matches the visual importance of the subband, subband coding leads to good quality image reconstruction and does not produce blocking artifacts.

Subband coding can be implemented as uniform frequency decomposition or non-uniform frequency decomposition. It is believed that non-uniform frequency decomposition may match the characteristics of the human visual system. A typical non-uniform frequency decomposition of subband coding is the Wavelet Transform, which was proposed by Mallat [23], Gharavi and Tabatabai [24], and Tran *et al.* [25] for coding of images. The Wavelet Transform decomposes the input image into basis functions which are dilations and translations of a single prototype function so-called mother wavelet. Each basis function at each scale of the wavelet decomposition is a different band, and each band is quantized with a different quantizer. To simplify the description of the Wavelet Transform, transform of one-dimensional signal is considered and then it is extended to two-dimensional signal.

Analogous to the transform coding schemes, for a given one-dimensional function $x(t)$, the Wavelet Transform is defined as the inner product of the function $x(t)$ itself and the wavelet function $\psi_{ab}(t)$ that is

$$\begin{aligned}(W_\psi x)(a,b) &= \langle x, \psi_{ab} \rangle \\ &= \int_{-\infty}^{+\infty} x(t)\psi_{ab}(t)dt \end{aligned} \tag{2.16}$$

and

$$\psi_{ab}(t) = \frac{1}{\sqrt{a}}\psi\left(\frac{t-b}{a}\right), \qquad \text{for } a > 0, b \in \mathcal{R} \text{ (i.e., real number).} \tag{2.17}$$

Thus the Wavelet Transform can be expressed as

$$(W_\psi x)(a,b) = \frac{1}{\sqrt{a}}\int_{-\infty}^{+\infty} x(t)\psi\left(\frac{t-b}{a}\right)dt \tag{2.18}$$

where the mother wavelet, $\psi(t)$, is localized and oscillating. The localization property means that it decreases rapidly to zero when the real variable t tends to positive or negative infinity. The oscillating property implies that the function vibrates like a wave; furthermore the integral of $\psi(t)$ is zero so as the first m movements of ψ. This can be expressed as

$$\int_{-\infty}^{+\infty} \psi(t)dt = \ldots = \int_{-\infty}^{+\infty} t^{m-1}\psi(t)dt = 0. \tag{2.19}$$

Daubechies [26] extended the above continuous transform for digitized signal and the Discrete Wavelet Transform (DWT) is applied by replacing the dilation or scaling factor, a, and the translation factor, b, by the dyadic values $\{a = 2^{-j} \text{and } b = k2^{-j}\}$. The wavelet functions become

$$\psi_{jk}(n) = 2^{j/2}\psi\left(2^j n - k\right) \tag{2.20}$$

The digitized signal $x(n)$ can be decomposited to the following linear combination of scaling functions and wavelet functions.

$$x(n) = \sum_{j=l}^{J}\sum_{k} d_j(k)\psi_{jk} + \sum_{k} c_J(k)\varphi_{Jk} \tag{2.21}$$

where φ_{Jk} is the scaling function and is defined as $\varphi_{Jk} = 2^{J/2}\varphi\left(2^J n - k\right)$; it is also known as a low pass filter while the wavelet function, ψ_{jk}, is a high pass filter. The coefficients $c_J(n)$ and $d_j(n)$ correspond to the approximation of $x(n)$ up to scale 2^J and the details of $x(n)$ (or the information lost) when the approximation of $x(n)$ up to scale 2^j goes to a coarser approximation with scale up to 2^{j-1}, respectively. An approximation and detail of the $(j-1)$th scale may be computed from the approximation of the jth scale by a recursive manner as shown in Figure 2.9. The filters are simply denoted as low pass, $g(k)$, and high pass, $h(k)$, filters to present a general form. The high pass filter is usually considered as the mother wavelet and the outputs of the high pass filters are thus the wavelet coefficients. In the case of two-dimensional signal, the one-dimensional scaling and wavelet functions $\varphi(x)$ and $\psi(x)$ yields one separable two-dimensional scaling function $\varphi(x)\varphi(y)$, and three separable two-dimensional wavelet functions $\varphi(x)\psi(y)$, $\psi(x)\varphi(y)$, and $\psi(x)\psi(y)$. The scaling function captures the low-frequency information

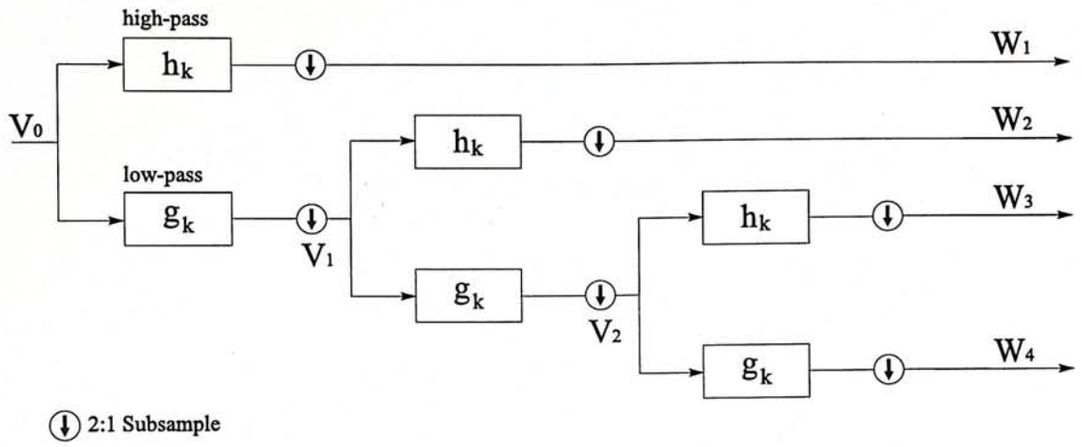


Figure 2.9: *Recursive subband of wavelet transform.*

while the wavelet functions capture the high-frequency information for various orientation.

2.3 Quantization

For all the spatial processing techniques discussed above, no compression can be achieved if all the parameters generated from the process are retained. As mentioned before, only a small fraction of the parameters contain perceptually important content. Quantization can be utilized to map these important parameters into a small but efficient set of data for compression. Quantization can be divided into two categories: *scalar quantization* and *vector quantization*. The following sections describe them separately.

2.3.1 Scalar Quantization

In scalar quantization, each input element is quantized at a time, and the output value depends only on that input. Such quantizers are useful in coding techniques such as predictive coding and transform/subband coding. There are two types of quantizers called *uniform quantizer*, which has a constant stepsize, and *non-uniform quantizer*, which has a variable stepsize. Both of them may also include a *dead zone* (i.e., the enlarged region that gets quantized to the level zero) to remove the noise-like perturbations around zero.

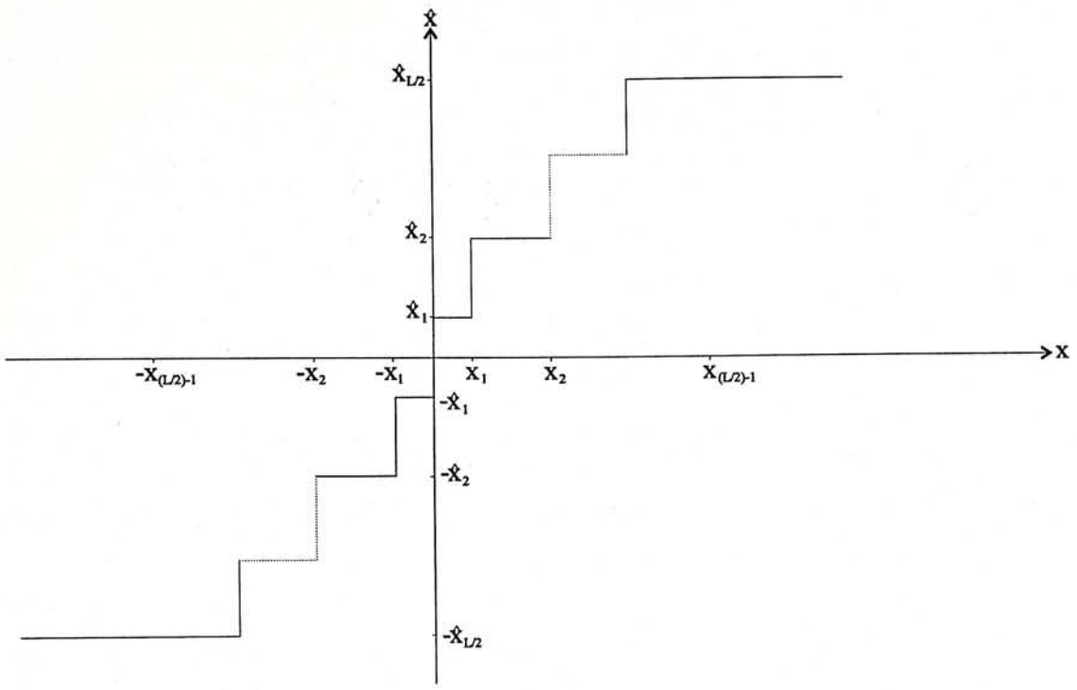


Figure 2.10: *Quantization function.*

The design of an optimal quantizer requires both the optimization criterion, such as the minimization of the mean squared quantization error (i.e., $E(x-\hat{x})^2$) , and the input probability density function, $p(x)$. Figure 2.10 shows a typical quantization function in which x is the input value, $\hat{x}$ is the quantized output value, and L is the number of quantized output values for the quantizer. In 1960, Max [27] showed that if $p(x)$ is an even function the conditions for minimal error are the following three equations.

$$\int_{x_{i-1}}^{x_i} (x-\hat{x}_i)p(x)dx = 0, \qquad i = 1,2,\ldots,\tfrac{L}{2} \tag{2.22}$$

$$x_i = \begin{cases} 0 & \text{for } i = 0 \\ \frac{\hat{x}_i+\hat{x}_{i+1}}{2} & \text{for } i = 1,2,\ldots,\frac{L}{2}-1 \\ \infty & \text{for } i = \frac{L}{2} \end{cases} \tag{2.23}$$

and

$$x_{-i} = -x_i \qquad \hat{x}_{-i} = -\hat{x}_i. \tag{2.24}$$

The quantizer that satisfies equations (2.22), (2.23), and (2.24) is called an L-level *Lloyd-Max* quantizer. 11 years later, in 1971, O'Neil [28] pointed out that a variable-length coded optimum uniform quantizer provides a lower code rate for a Laplacian $p(x)$

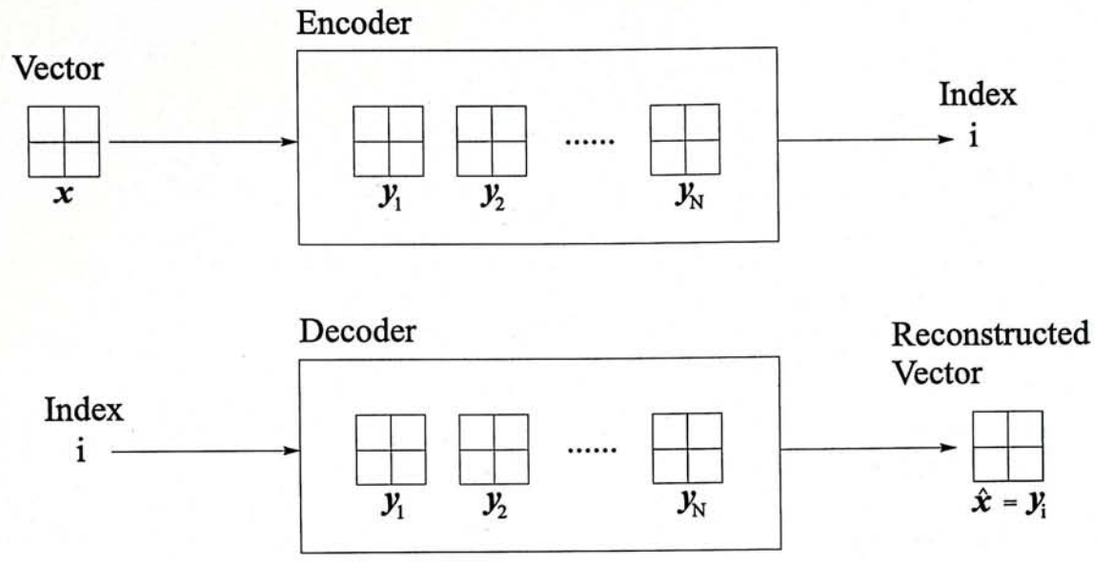
than a fixed-length coded Lloyd-Max quantizer with the same output fidelity. His work revealed the close relationship between quantization and code word assignment. MPEG makes use of this point, where uniform quantizer and variable-length coding are applied. This is elaborated in the appendix attached.

For quantizing transform/subband coefficients, the stepsize for each coefficient can be fixed but the relative stepsizes of the quantizers for the different coefficients should be varied to match the differing perceptual importance of the various coefficients. The human visual system is less sensitive to high frequency. As a result, high frequency coefficients are usually quantized more coarsely than low frequency ones. To simplify the application of different stepsizes, a weighted quantization matrix can be defined to normalize coefficients before quantizing them.

2.3.2 Vector Quantization

The basic idea of vector quantization is to compress a group of pixels, a group of transform coefficients, or any other group of information jointly, instead of one at a time. By operating directly on $m \times m$ groups (or vectors), quantizers map them onto a set of finite reproduction vectors known as the *codebook*. The mapping operation is carried out by pattern matching between the input vector and the codebook entries. The index for each mapping is transmitted to represent each vector. The decoder uses the index to look up the corresponding code vector in the codebook and inserts it into the reconstructed image. The typical block diagram for vector quantizer is depicted in Figure 2.11. From the figure, vector quantization can be seen as a combination of two functions: an encoder that views the input vector $\boldsymbol{x}$ containing M elements and generates the index of the reproduction vector specified by i and a decoder that uses this index to generate the reproduction vector $\boldsymbol{y}_i$ from a set of N code vectors. As indices always contain fewer bits than the original vectors, compression is achieved.

The matching criterion can be defined mathematically by the minimization of a distortion



$y = \{ y_1, y_2, \cdots, y_N \}$ is the set of reproduction vectors called codebook

Figure 2.11: *Block diagram of VQ.*

measure, $d(\boldsymbol{x}, \boldsymbol{y}_i)$, which can be expressed as

$$\begin{aligned} d(\boldsymbol{x}, \boldsymbol{y}_i) &= \|\boldsymbol{x} - \boldsymbol{y}_i\| \\ &= \sum_{k=1}^{M} (x_k - y_{i,k})^2, \end{aligned} \tag{2.25}$$

where $\|arg\|$ denotes the Euclidean norm, and x_k and $y_{i,k}$ are the kth element of vectors $\boldsymbol{x}$ and $\boldsymbol{y}_i$, respectively. With equation (2.25) the optimal vector quantizer can be obtained by minimizing

$$\begin{aligned} D &= E\{d(\boldsymbol{x}, \boldsymbol{y}_i)\} \\ &= E\{d(\boldsymbol{x}, Q(\boldsymbol{x}))\} \\ &= \int d(\boldsymbol{x}, Q(\boldsymbol{x}))\, f(\boldsymbol{x}) d\boldsymbol{x}, \end{aligned} \tag{2.26}$$

where $\boldsymbol{y}_i$ is expressed as a function of $\boldsymbol{x}$ (i.e., $Q(\boldsymbol{x})$), and $f(\boldsymbol{x})$ is the probability density function (pdf) of the input vectors. Linde *et al.* [29] suggested an algorithm called LBG algorithm to generate locally optimal vector quantizers by choosing $\boldsymbol{y}_i$ to be the centroid of its M dimensional cell and satisfying a *nearest neighbor* rule, which can be expressed

as choosing $\boldsymbol{y}_i$ if and only if $d(\boldsymbol{x}, \boldsymbol{y}_i) \leq d(\boldsymbol{x}, \boldsymbol{y}_j)$, for $i \neq j$ and $1 \leq j \leq N$. This algorithm has two disadvantages: computation cost is high and globally optimal design is not guaranteed. Tree-structured codebooks [30] can be applied to reduce search time. The basic idea is simply to perform a tree search on a structured codebooks to replace the exhaustive searching of unstructured codebooks. Although this method effectively reduces the search time, it requires double memory size for the codebook storage. Other methods such as lattice VQ [31] are proposed to deal with the memory problem.

In spite of the complexity of the encoder and computational cost, vector quantization facilitates its application for video broadcast since only the design of decoder (i.e., receiver) is concerned, which only requires simple codebook lookup operations.

2.4 Code Word Assignment

Following quantization, an efficient set of symbols is needed to be created to represent the quantized video data. As mentioned in the chapter 1, it is the work of code word assignment to produce a digital bitstream for transmission or storage with reduced average bits per symbol. If the quantizer output is simply coded by a fixed-length binary code word having B bits, the resultant code is not optimal. It is because quantized symbols have different probabilities. Shannon [3] told us that there exists a code that uses less than B bits per symbol for this situation. Entropy coding is often utilized to assign code words to quantized symbols. Its aim is to encode the N set of symbols with probabilities p_i, for $i = 1, 2, \ldots, N$, by $-\log_2 p_i$ bits, so that the average bit rate is the entropy H, which is defined as

$$H = -\sum_{i=1}^{N} \log_2 p_i. \tag{2.27}$$

The most popular algorithm for this purpose is Huffman coding [1]. Huffman's approach is to generate a codebook by three steps [32]:

1. The symbols are arranged according to their probabilities, p_i, in a descending order and they are considered as a leaf nodes of a tree.

2. If there is more than one symbol in this stage of source reduction, the two lowest probability symbols are combined into a single symbol that replaces them as a single node in the next source reduction. The probability of this new node is the sum of the two combined symbols. Each pair of branches that combine into a node is assigned "1" and "0" arbitrarily. A series of source reduction is carried out by repeating step 1 and 2 until a reduced source with only two symbols is reached.

3. The code word for each original symbol is obtained by reading the "1" and "0" assigned to each branch sequentially from the root node (the smallest source) to the leaf node (the original source).

Following the generation of the codebook, coding and decoding is done simply by codebook lookup. Since the above process is computationally complex for large number of symbols, sacrificing coding efficiency for simplicity is possible by modifying the Huffman code.

Truncated Huffman code is one of the practical versions of Huffman code, where the most probable N_1 symbols for $N_1 < N$ are coded by Huffman coding while the remaining $N - N_1$ symbols are coded by appending a prefix code in front of a fixed-length code.

Another practical version of Huffman code is known as the *Huffman shift code*, where the N symbols are divided into blocks with N_1 symbols. Thus $q = int\left(\frac{N}{N_1}\right)$ blocks of symbols and the remaining j symbols, for $0 \leq j \leq N_1 - 1$, are obtained. The first block consisting the most probable N_1 symbols is coded by Huffman coding while the remaining blocks are coded by a prefix code representing the block number followed by the Huffman code of the N_1 symbols in the block. To simplify the algorithm, the same Huffman codebook is usually used for each block. The remaining j symbols are treated as a block with j symbols. Normally, the prefix code for the second block is assigned to be one of the shortest Huffman code words in the first block. The prefix code for the third block consists two prefix codes for the second block and so on, i.e., if "00" is the prefix code for the second block, "00 00" will be the prefix code for the third block and

"00 00 00" will be the prefix code for the fourth block and so on.

Another coding technique used for code word assignment is run-length coding, which codes the number of zeros between two successive non-zero values, i.e., the length of the runs of zeros are coded. It is especially useful to code the quantized coefficients after transform/subband coding, where most of the coefficients are quantized to zero. Both CCITT recommendation H.261 and MPEG utilize an entropy coding similar to the truncated Huffman code discussed above to code a {run, amplitude} pair of the DCT coefficients. This will be investigated further in the appendix attached.

2.5 Selection of Video Coding Standard

Compatibility to the existing video system is an important beneficial factor towards the usefulness of a new video system. There are a number of digital video systems existing in the user community. The most popular digital video systems include: Indeo video, CCITT recommendation H.261 and ISO MPEG.

- **Indeo video** was introduced in November 1992 by Intel [33], which uses VQ-based algorithms for compressing and decompressing and is optimized for decompression on standard Intel microprocessors, rather than for maximum compression ratio or maximum image quality. It was first announced in a videoconferencing application called ProShare Video System 200. However, It is developed not only for video-conferencing application but also for various forms of digital video applications including Microsoft's Video for Windows, Apple's QuickTime, and IBM's Multi-media Presentation Manager. As Indeo video is optimized for individual frame quality, it drops coming frames and shows the previous frame whenever the data rate exceeds an allowable limit. That is why temporarily freezing videos are obtained for playing video application on windows environment. Davis [33] indicated that Indeo playback rates vary with processor which is shown in Table 2.1.

	640 × 480	320 × 240	160 × 120
i486SX/25	1 frame/sec	15 frame/sec	30 frame/sec
i486DX2/66	10 frame/sec	30 frame/sec	30 frame/sec
Pentium	20 frame/sec	30 frame/sec	30 frame/sec
i750 coprocessor	30 frame/sec	30 frame/sec	30 frame/sec

Table 2.1: Indeo Playback Rates Vary with Processor

- **CCITT recommendation H.261** [34] was announced in 1990 by the international standards body (International Committee on Telegraph and Telephones), which was started by the CCITT Specialist Group XV in 1984. It is based on DCT algorithm to reduce spatial redundancy and motion compensation to reduce temporal redundancy and is developed especially for videoconferencing and videophone applications with a bit rate range of $p \times 64$ kbits/sec, for $1 \leq p \leq 30$, i.e., from 64 kbits/sec to 1.92 Mbits/sec. Unlike Indeo video, H.261 specifies only two fixed image sizes which are the common interchange format (CIF) with 352×288 pixels and the quarter CIF (QCIF) with 176×144 pixels. Being optimized for real time telephony applications, H.261 is developed by minimizing the encoding and decoding delay while maintaining a fixed data rate.

- **MPEG video** was adopted as an international standard in 1992. Like CCITT recommendation H.261, it is a DCT-based coding algorithm but extends motion estimation to bidirectionally interpolated image frames. MPEG video supports various resolutions and image sizes. Being designed to maintain picture quality with maximum compression rather than to minimize coding delay, it is suitable for non-real time applications such as electronic publishing, video games and delivery of movies where maximizing both picture quality and compression ratio are preferred.

Considering that Indeo video may generate temporarily freezing and it is not an international standard, both CCITT recommendation H.261 and MPEG video are superior to it. Further, stereoscopic video is well suited for applications such as video games and delivery of movies which the design of MPEG targeted. Therefore, we design our

stereoscopic video coding scheme to be compatible with MPEG video systems.

Chapter 3

MPEG Compatible Stereoscopic Coding

3.1 Introduction

Stereoscopic imaging is different from true three-dimensional imaging that requires holography technique, i.e., the reconstruction of the wavefront of the objects in a scene. True three-dimensional images not only provide the sensation of depth, but also allow observers to "look around" to their sides and perhaps even their back. Although Hilaire *et al.* [35] showed that real-time "holographic video" is possible, it is still a long way from bringing it to the general public. Stereoscopic imaging, instead, is easier to generate and has a longer history. Stereoscopic imagery dates back to the invention of the anaglyph technique, which requires red/green glasses as selection devices for the left and right eyes. Sand [36] pointed out that a successful experimental program produced by the German broadcasting organization Norddeutscher Rundfunk (NDR) in 1982 titled "When Television Pictures Become Three-Dimensional" using this technique found extremely high public response. However, the production of full color stereoscopic images is not possible via this technique. Other approaches including the application of polarized glasses as selectors, LCD shutter glasses and autostereoscopic systems, such as the application of lenticular sheets, are developed. Besides, head-mounted display that was first introduced

by Sutherland [37] is also the stereoscopic display technology widely used in virtual reality systems.

The principle behind glasses wearing stereoscopic systems is to occlude one eye when the image for the other eye is displayed and vice versa. In the lenticular sheets approach [38], images are displayed on an array of half cylindrical lenses (lenticular sheets) as a sequence of narrow vertical stripes, left-eye image, right-eye image, left, right, etc., roughly matching the pitch of the lenses. If the observer is positioned in exactly the right place, the left image is refracted into the left eye while the right image is refracted into the right eye after passing through the lenticular sheets. An authentic stereo image can thus be observed.

All the techniques including head-mounted display described above depend on the reconstruction and the transmission of the left and right channels of video sequence. They replicate the way we naturally view our surroundings as our sense of depth comes from seeing objects with two eyes, each with a slightly different view point. Stereopsis, which is a process performed by our brain, fuses these two images into one that has depth. A technique delivering the two parts of a stereoscopic video sequence via conventional TV broadcast channel was proposed by Lipton [39]. It squeezes the two parts of a stereoscopic video sequence into a side-by-side format. At the receiver, a demultiplexing circuitry is required to turn the image pairs into a field-sequential stereoscopic display and trigger an active LCD glasses. Although this method makes the delivery of stereoscopic video using conventional broadcast equipment possible, the resolution of the resulting images is degraded as only one of odd and even fields would be seen.

Using the video coding technique discussed in the previous chapters, the two channels of video sequence can be coded independently forming two separate bitstreams. Gomi *et al.* [40] proposed such a prototype system based on DCT algorithm to transmit and present full color stereoscopic video sequence via ISDN. However, only the spatial compression

technique was used. Neither the interframe prediction nor the correlation between the left and right parts of the video sequence were applied. Even if the interframe prediction is utilized, comparing to a single channel video codec, the transmission and/or storage require twice the bandwidth and/or storage capacity, which keep the stereoscopic video systems from widespread application. As mentioned previously, the two channels of video sequence simulate the views from two eyes, each having a slightly different view point at the same instants from a scene, they contain many similarities. Yamaguchi *et al.* [41] confirmed this point by studying the statistical characteristics of a stereo image pair. The generation of a more compact stereoscopic video bitstream is possible by removing these similarities. Ziegler *et al.* [42] reported the starting of a research and development group aiming at developing a hardware prototype of a stereoscopic system for broadcast and non-broadcast applications. The covering area of the project is very broad, including 3DTV-camera, stereo display and building a digital codec and multiplexing/demultiplexing for transmission over an IBCN-channel, and test & evaluation. The coding strategy proposed utilizes both interframe and disparity information between the two channels. In addition, the compatibility of MPEG had also been concerned. However, the proposed codec is quite complex and no actual result is reported so far.

In the next section, MPEG compatibility is defined. Then approaches leading to MPEG compatible stereoscopic bitstream are discussed.

3.2 MPEG Compatibility

There are at least two forms of compatibility that can be defined. They are compatibility of the coder and decoder, and compatibility of the bitstream syntax. Compatibility of the coder and decoder means that existing single channel MPEG coder and decoder are embedded in the stereoscopic coder and decoder, respectively. On the other hand, compatibility of the bitstream syntax means that the syntax of the stereoscopic bitstream generated is a superset of the syntax of the monoscopic bitstream with both downward

and upward compatibility.

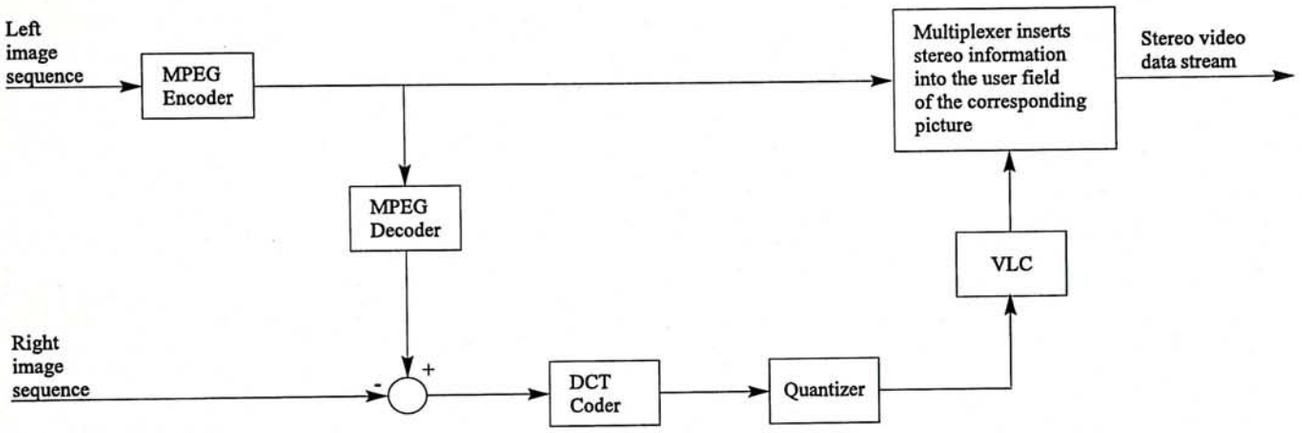
MPEG defines a user data field for specific application. By making use of this data field, a user data stream is allowed to be embedded within the MPEG decodable bitstream. Hence, one of the channels of a stereo video sequence could be coded directly to MPEG bitstream while the stereoscopic information is inserted in this user data field. The bitstream so generated is bitstream compatible with a single channel MPEG bitstream and therefore is decodable by a single channel MPEG decoder.

3.3 Stereoscopic Video Coding

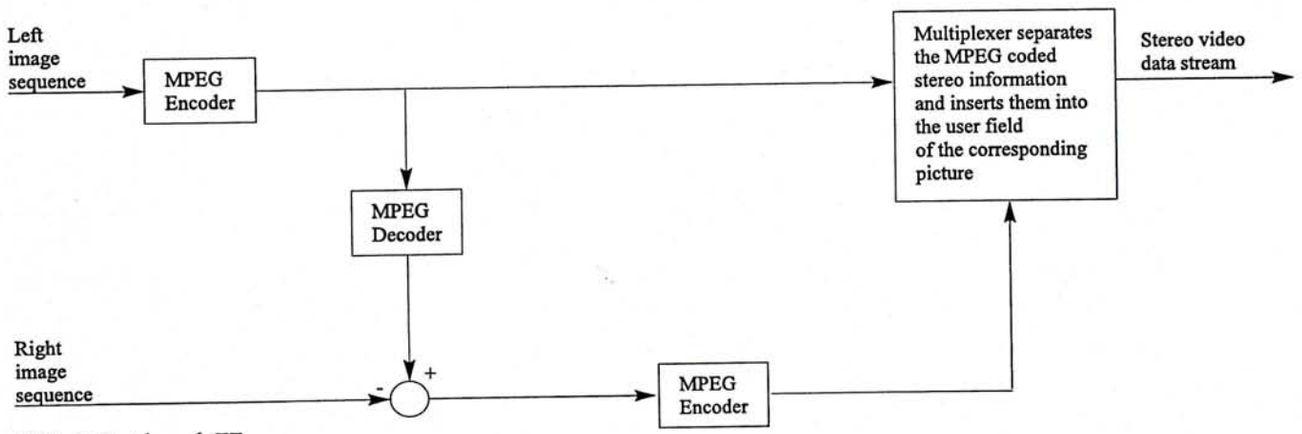
Having observed that one of the channels of stereoscopic video sequence must be coded directly to MPEG bitstream to maintain MPEG compatibility, further compression is not achievable in this channel and the design of an efficient stereoscopic video encoder becomes the main objective for generating efficient representation for the stereoscopic information. The simplest approach is to find the stereoscopic information as the differences between each stereo image pair. We simply call it "coding by stereoscopic differences".

3.3.1 Coding by Stereoscopic Differences

In this approach, each image frame in one channel is subtracted directly from the other. All the common area in the two channels of video sequence is eliminated. The remaining differences can then be coded by other methods and the coded data are inserted into the user data field. We tried two methods to code the difference sequence. Figure 3.1 shows the main function block of the stereoscopic coding by differences. The left image sequence is first coded to single channel MPEG bitstream. Before passing to a multiplexer, the bitstream is duplicated and feeds to an MPEG decoder. The output of the MPEG decoder is a sequence of the reconstructed image frames which is subtracted by the corresponding right image frame to produce image differences. In the first method as shown in Figure



(a) Method I.



(b) Method II.

Figure 3.1: *Stereoscopic coding by differences.*

3.1a, those image differences are DCT coded to form transform coefficients. The transform coefficients are then quantized and pass into a variable length coder (VLC). The huffman coded coefficients output from the VLC feed into the multiplexer in which the coded coefficients are inserted into the user data field of the corresponding coded pictures in the MPEG bitstream of the left channel. In the second method as shown in Figure 3.1b, the image differences are fed into another MPEG encoder to code them as MPEG bitstream. This bitstream is decomposited into individual coded pictures and inserted into the user data field of the corresponding coded pictures in the MPEG bitstream of the left channel by the multiplexer.

These methods were simulated on computer. In the first method 8×8 DCT operation was used to code the stereoscopic differences. In addition, the same quantizer step size as the default 8×8 frequency dependent quantization matrix and VLC code used for the coding of I-pictures defined by MPEG were used to implement the quantizer and the variable length coder. In the second method the same MPEG encoder coded for the left channel was used to code the stereoscopic differences. It is obvious that for the extreme case with exactly the same left and right image sequences which simulates the case for observing very far scene, 50% further compression comparing to two independent normal MPEG coded bitstreams was obtained by these methods. However, in this extreme case, no disparity is actually observed, and therefore no binocular depth perception is conveyed. In the case with typical stereoscopic view which is generated from slightly different viewpoints, each image pair has some differences. Coding by stereoscopic differences gave worse results comparing to two independent normal MPEG coded bitstreams. It is because, for the first method, the stereoscopic differences are coded frame by frame and only spatial redundancy is exploited. For the second method, although both spatial and temporal redundancies are exploited by MPEG encoder, the difference images generated contain more AC signals comparing to the corresponding original images. Thus, the DCT coefficients of the difference images require more bits to obtain comparative quality from normal MPEG code of the original images. Figure 3.2 shows a sample image pair

and the difference image generated by the image pair. The AC signals in the difference image are observed as random noise. It is especially serious at the window in Figure 3.2. This makes the difference image more difficult to code comparing to the original right image.

3.3.2 I-pictures only Disparity Coding

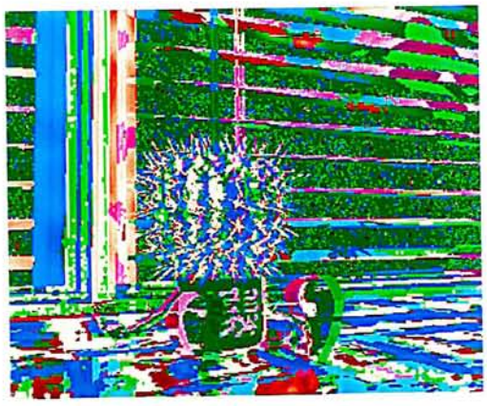
Alternative ways to represent stereoscopic information are sought by reviewing the elimination of the similarities between still stereo image pairs. Both disparity compensation and three dimensional Discrete Cosine Transform (3D-DCT) were proposed by Dinstein *et al.* [43]. The method of 3D-DCT is to consider the pair of 2D image signal as a set of 3D image signal waveform. However, MPEG bitstream consists of two dimensional DCT coefficients. Using 3D-DCT will destroy the backward compatibility. Therefore, disparity compensation is utilized.

Before going to the detail of disparity compensation, the term disparity must be defined. Stereo disparity is a physiological term coming from the difference in angle between the viewing axes of the left and right corresponding points in the two images taking from the left and right eyes. The term "disparity" as used in the following discussion is defined as the distance in term of pixel unit between similar objects in a stereo image pair when the two views are superposed such that the image boundaries are aligned.

By using disparity compensation, the generation of stereoscopic video bitstream can be achieved by the transmission of only one channel plus a disparity information signal. This concept was first implemented by Lukacs [44] in 1986, who introduced a technique called *Disparity Corrected Prediction*. In his approach, each left image can be used to reconstruct a right image using disparity corrected prediction. However, it may not regenerate image frames better than those via interframe prediction. Actually, the differences between a stereo image pair are usually large compared to those between successive frames for interframe coding. Furthermore, they do not decrease when there is no motion in the

Left image

Right image



Difference image

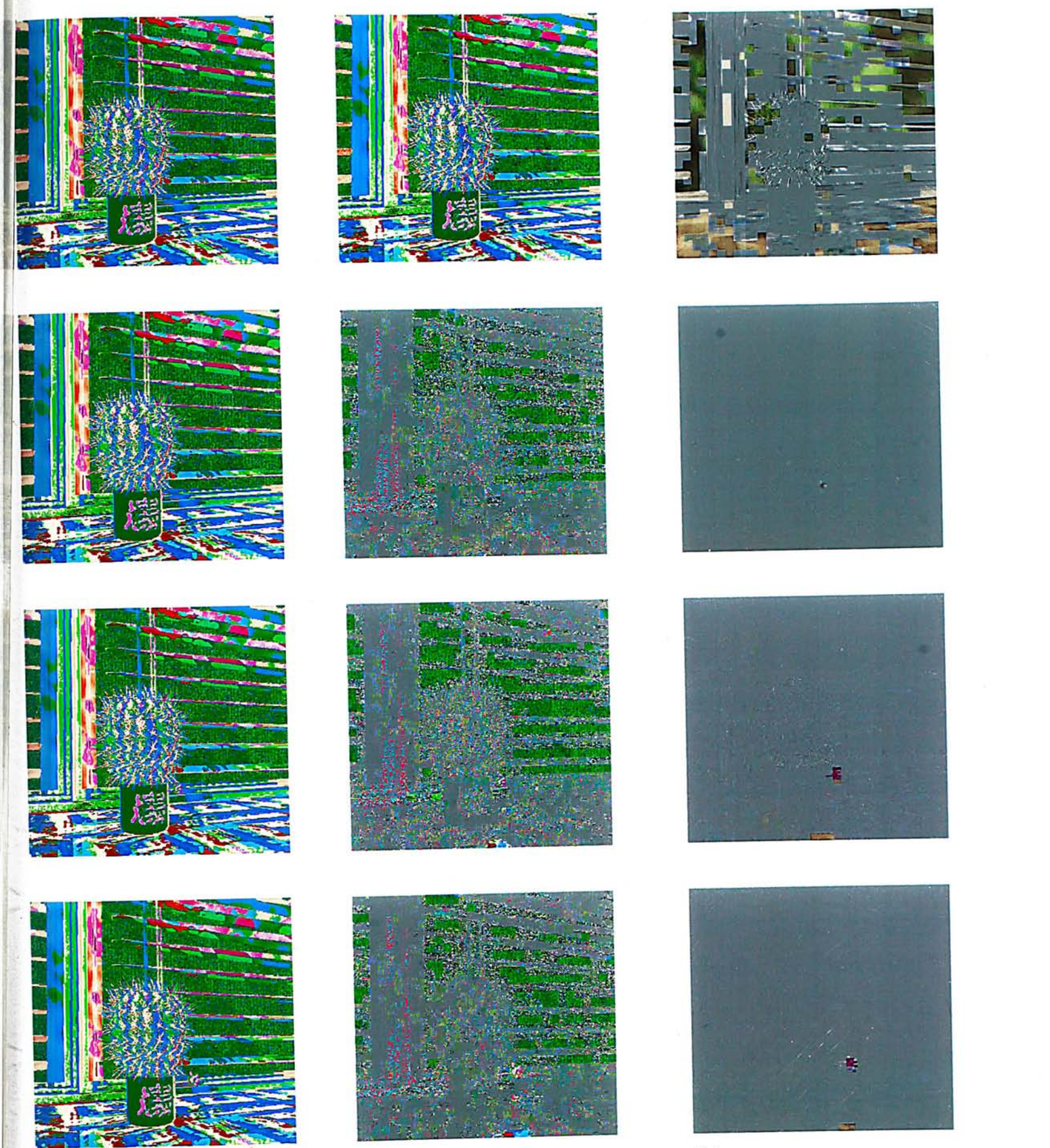
Figure 3.2: *Sample image pair (top) and the difference image generated (bottom).*

local area as their source is due to "disparity" instead of motion.

In the I-pictures only disparity coding scheme, interframe prediction is reserved and disparity corrected prediction is applied to I-pictures, which have largest sizes among the MPEG picture types. This scheme was reported at the *Third International Symposium on Consumer Electronics* [45]. It makes use of the advantages of interframe prediction to reduce temporal redundancy and disparity corrected prediction to utilize the correlations between the two channels of video sequence. Interframe prediction incorporating motion compensation is processed as normal MPEG coding while disparity corrected prediction requires to find disparities on each stereo I-picture pair. This can be done by separating all objects in the scene at different depths. However, it involves complicated image processing technique. A simpler approach had been taken to partition the right image frame into subimage sections (blocks) and each block is considered as an object. Block matching between the right image frames and left reference frames is performed to compute disparities of each right image block. Due to human physiology that human eyes are displaced horizontally, only horizontal dimension requires to be dealt with.

The differences between this approach and the previous one can be summarized by Figure 3.3. In the figure, the gray blocks represent areas which are predictable and the color blocks are residual after prediction. For the previous approach, the first method applying DCT coding to encode the sequence shown in Figure 3.3(a) which contains a lot of image contents while the second method applying DCT coding to encode the sequence shown in Figure 3.3(b) which contains less image contents comparing to those for the first method. In this approach, however, the sequence shown in Figure 3.3(c) which contains the least image contents is to be encoded.

The following section details the algorithm of the stereosopic MPEG encoder which deals with the disparity of the I-pictures. The scheme bases on building blocks which compose of MPEG encoding and decoding hardware to ease the realization of the codec.



(a) For method I

(b) For method II

(c)

Coding by Stereoscopic Differences

I-picture only Disparity Coding

Figure 3.3: *Sequences of stereoscopic differences to be coded for different approaches.*

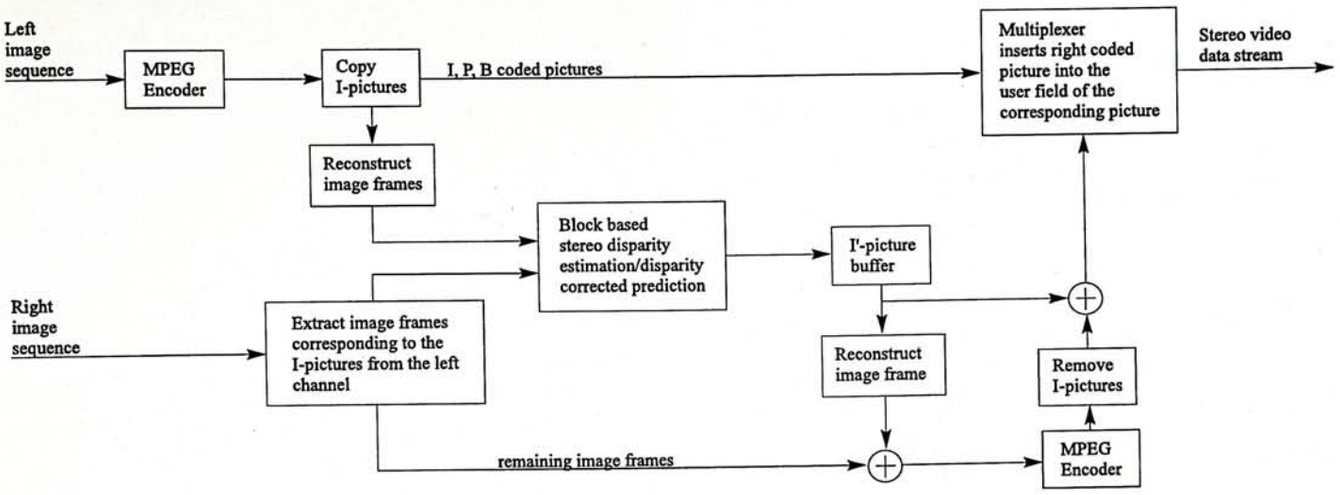


Figure 3.4: *Stereo MPEG coding scheme by I-pictures disparity coding.*

3.4 Stereoscopic MPEG Encoder

Figure 3.4 shows the main function block of the MPEG compatible stereoscopic coding scheme. The left source picture sequence is encoded as normal MPEG bitstream. The I-pictures in the left MPEG bitstream are reconstructed back into image frames and along with the corresponding right image frames forming stereoscopic pairs. They are then passed to the stereo disparity estimator in which the right images are divided into 16×16 image blocks. Disparity vector as well as the residual are then computed by operating on those image blocks. A control signal is also generated to determine whether good or bad match is achieved. For good match, the disparity vector and the residual are passed to an I'-picture buffer. On the other hand, for bad match, the right image block is simply compressed by DCT and passed to the same buffer. The operation of the stereo disparity estimator is detailed in the next section.

After the processing for all blocks in a complete frame, the contents of the I'-picture buffer are ready. The I'-picture is then duplicated and reconstructed back into image frame. The image frame as well as the rest of the right image frames forms a new right image sequence which is passed to a normal MPEG encoder. The I-pictures of the resulting right MPEG bitstream are removed and replaced by the I'-pictures stored in the buffer.

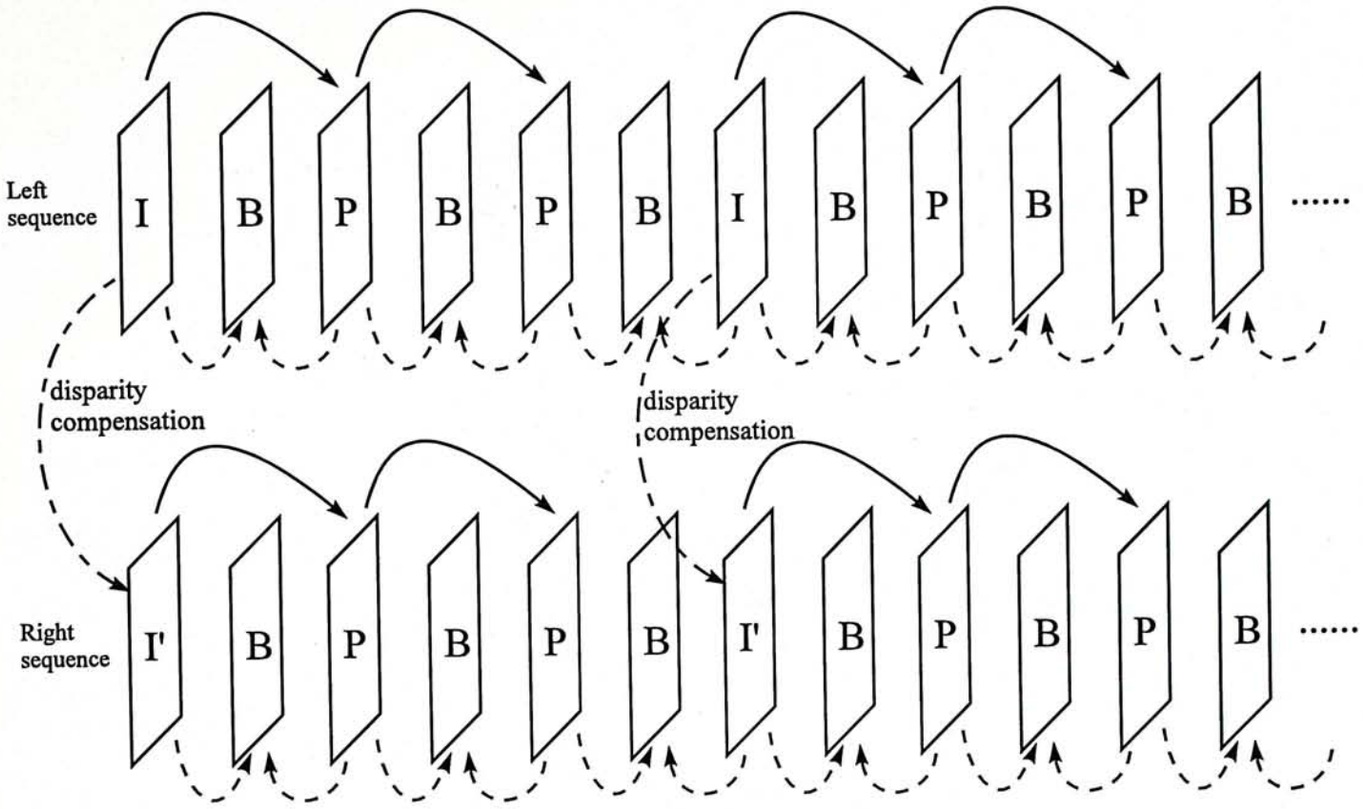


Figure 3.5: *Prediction of stereoscopic image sequence.*

The edited right bitstream is further multiplexed into the user data field of the left video bitstream forming a stereo MPEG video bitstream.

Figure 3.5 depicts the prediction of a complete stereoscopic image sequence, in which the I-picture of the left part of a stereoscopic video sequence plays three roles. It is a reference for predictive (P), bi-directional predictive (B), and disparity compensated prediction for the corresponding right image frame.

3.4.1 Stereo Disparity Estimator

The stereo disparity estimator is a block based operator. Firstly, it divides the input right image frames into 16×16 image blocks. Each image blocks is compared with the corresponding left image frame horizontally moving from pixel to pixel to find a best match, hence the disparity vector. A cost function is used for the search which gives the total absolute error for each match. This function is similar to the mean absolute error

described in equation (2.1) without taking the mean. The function is defined as follows:

$$E(j) = \sum_{m=1}^{16} \sum_{n=1}^{16} |U_r(m,n) - U_l(m,n+j)|, \qquad -p \leq j \leq p. \tag{3.1}$$

where $U_r(m,n)$ and $U_l(m,n)$ are the luminance value of right and left image block at row m and column n, respectively and p defines the search area in the left image. The value j that gives the minimum $E(j)$ along the search area is the best match disparity vector of the block being coded. Accordingly, the best match can be found for each block. However, it only specifies that the best approximation of the block being coded is found in the left image frame instead of the exact match. If the left and right best match block pair is not deviated too much, the matching is classified as a good match which means that disparity compensation is successful for this block. The disparity vector is coded, and the DCT coefficients of the disparity compensation error (i.e., the differences between the left and right best match block pair) are Huffman coded. They are then transmitted to replace the block being coded. If the left and right best match block pair is deviated too much, the matching is classified as a bad match which means that disparity compensation for this block is failed. The coded disparity vector and the coded DCT coefficients of the disparity compensation error are not as efficient as the Huffman coded DCT coefficients of the original block being coded. Thus, the coded DCT coefficients of the original block are transmitted. Whether the best match is a good match or bad match can be determined by the comparison between the variance of the disparity compensation error (i.e., right image block - best match reconstructed left image block), ${\sigma_d}^2$, and the variance of the right image block itself, ${\sigma_r}^2$.

The variances are defined as the average of the square of the deviations of the elements in the block from the mean value. The term "elements" refers to the pixel difference for ${\sigma_d}^2$ and the pixels in the right image block being coded for ${\sigma_r}^2$. It can be seen that the larger the variance, the more bits are required to code the image block in order to maintain the same quality. Therefore, we define ${\sigma_d}^2 < {\sigma_r}^2$ as the condition for good match while ${\sigma_d}^2 \geq {\sigma_r}^2$ as the condition for bad match.

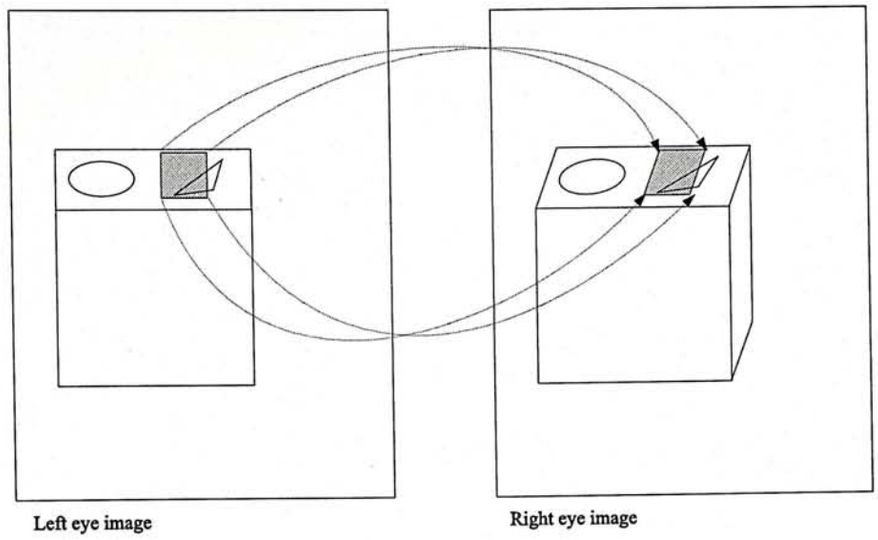


Figure 3.6: *The principle of improved disparity estimation.*

3.4.2 Improved Disparity Estimation

The block matching disparity estimation can be improved by applying *generalized block matching* technique which was introduced by Seferidis and Chanbari [46] to enhance motion estimation. It was first designed to handle the complex motion of objects by comparing each block of the current frame with a deformed quadrilateral of the previous one. When it is applied to stereo disparity estimation, only horizontal deformation is considered; this simplifies the operations. The approach is to find the mapping parameters of the function f which relates the coordinates of the corresponding pixels in the image pairs:

$$x_i^r = f(x_i^l, y_i^l) \text{ and } y_i^r = y_i^l \tag{3.2}$$

where (x_i^r, y_i^r) and (x_i^l, y_i^l) are the coordinates of the pixels in the right and left image blocks, respectively. The matching criterion is then applied on the transformed quadrilaterals given in equation (3.2) as shown in Figure 3.6. The mapping function may be any function but bilinear function, which is most commonly adopted for the interpolation of unknown pixel values from the surrounding ones, has been chosen for our work. The equation (3.2) becomes

$$x_i^r = \alpha_0 x_i^l + \alpha_1 y_i^l + \alpha_2 x_i^l y_i^l + \alpha_3 \text{ and } y_i^r = y_i^l \tag{3.3}$$

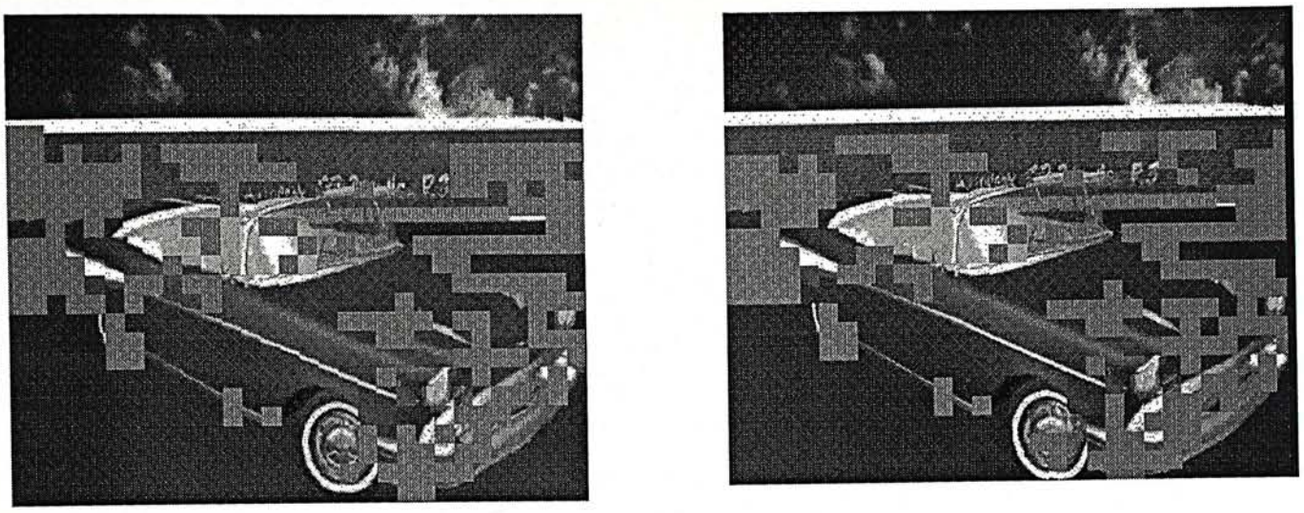


Figure 3.7: *The predicted image frames generated by stereo disparity estimation (left) and improved disparity estimation (right).*

The first stage of the estimation is a normal full search method. The disparity compensated image frame is then compared with the right eye image blocks. For those exhibit large differences, the second stage is performed with the improved block matching algorithm. Four equations are constructed according to equation (3.3) with the four corners of the original right eye image block and the deformed corresponding left image block. The four αs are solved by operating on the following matrix equations.

$$\begin{pmatrix} x_0^r \\ x_1^r \\ x_2^r \\ x_3^r \end{pmatrix} = \begin{pmatrix} x_0^l & y_0^l & x_0^l y_0^l & 1 \\ x_1^l & y_1^l & x_1^l y_1^l & 1 \\ x_2^l & y_2^l & x_2^l y_2^l & 1 \\ x_3^l & y_3^l & x_3^l y_3^l & 1 \end{pmatrix} \begin{pmatrix} \alpha_0 \\ \alpha_1 \\ \alpha_2 \\ \alpha_3 \end{pmatrix} \tag{3.4}$$

Figure 3.7 shows the predicted image frames generated by stereo disparity estimation and improved disparity estimation. The missing blocks are those blocks with bad match and are coded with intra mode. Figure 3.8 shows the enlarged views of the car. In this case, only four pixels horizontal search regions for each corners was applied. It is obvious that the saw-teeth artifact is removed by the improved disparity estimation. For better improvement, the search regions can be increased but this involve more computation. Since the better the predicted image frame, the smaller the error signal is and hence the transmission bit rate. Improved disparity estimation can improve the efficient of

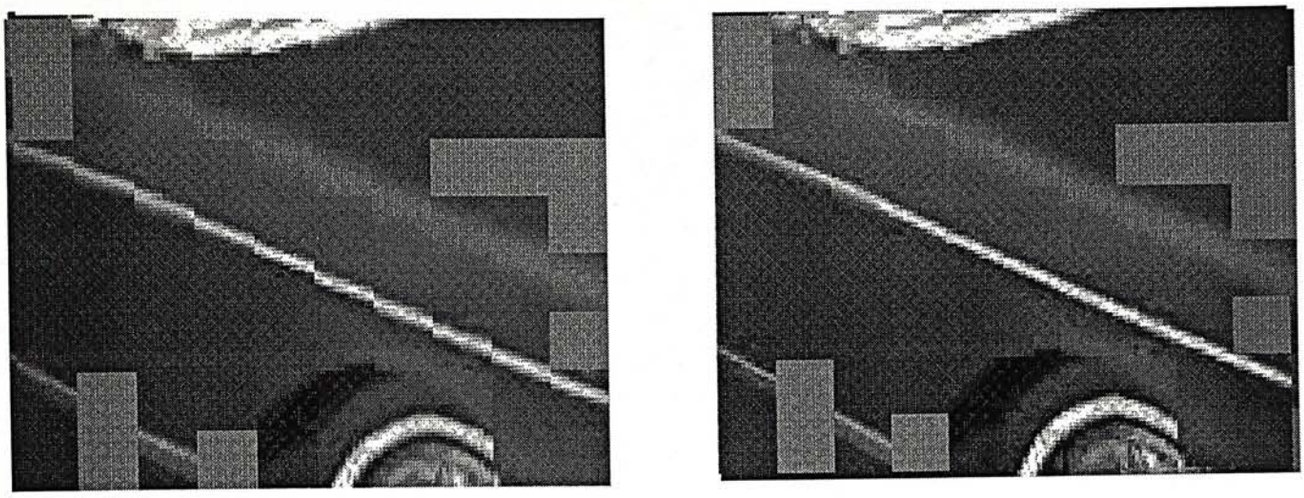


Figure 3.8: *The enlarged views of Figure 3.7.*

the compression. However, the improvements in both picture quality and compression efficiency are at the expense of computation time. Moreover, this technique involves four extra parameters, i.e., the four αs, to be transmitted for each block. As the parameters are floating point variables, they will complicate the bitstream syntax. Thus, in the following implementation no improved disparity estimation was applied.

3.4.3 Stereo Bitstream Multiplexer

The multiplexer identifies the coded pictures of the edited right bitstream. Each coded picture in the right bitstream is inserted into the user data field in the picture layer of the corresponding left picture so that bitstream in the picture layer contains the information of a stereo pair and this provides synchronization of the left and right image pair. As defined by MPEG the contents of user data may not contain any block with the "start code", which consists of 23 consecutive zero bits followed by an one bit. To prevent this, the stereoscopic information that is to be inserted as user data is scanned for the "start code". If it is found, a code word with 24 consecutive zero bits followed by an one bit is replaced. The extra zero bit will be removed at the decoder by a similar process. Figure 3.9 shows the data structure of stereo MPEG video bitstream.

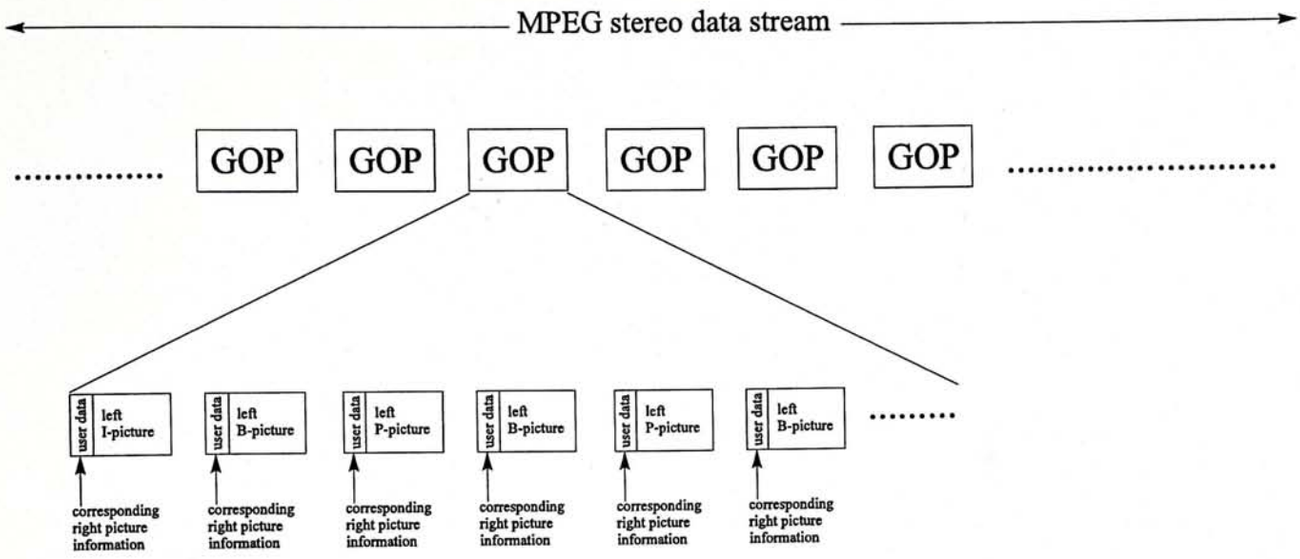


Figure 3.9: *Stereo MPEG data structure.*

3.5 Generic Implementation

Up to here, the scheme described bases on building blocks including complete MPEG encoders. Although this eases the realization of the stereo coding scheme, there are redundant operations existing in this arrangement. Referring to Figure 3.4 again, the left image sequence is coded by a normal MPEG encoder. Each I-picture in the resultant left channel MPEG bitstream is duplicated and decoded back into the corresponding image frame for stereo disparity compensation. These operations are redundant since there is a decoding path inside a normal MPEG encoder, which is shown in Figure 3.10. This decoding path is necessary to generate the anchors which are the reference image frames for motion compensation. A generic design utilizing this decoding path to get the decoded I-pictures of the left channel for the stereo disparity compensation process would provide better performance.

A block diagram of the generic design is shown in Figure 3.11. It includes the following: macroblock converters, DCT and inverse DCT operators, quantizers and inverse quantizers, variable length coders (VLC), frame storages (FS) for succeeding frame, preceding frame and left image frame, motion compensators, a stereo disparity compensator, and a

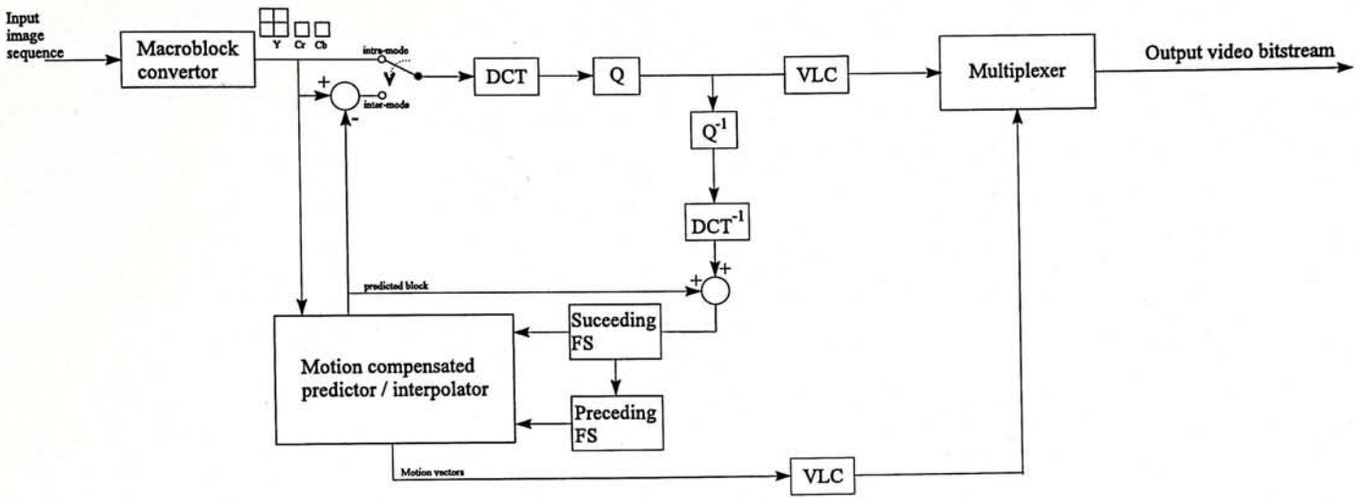


Figure 3.10: *Block operation of normal MPEG encoder.*

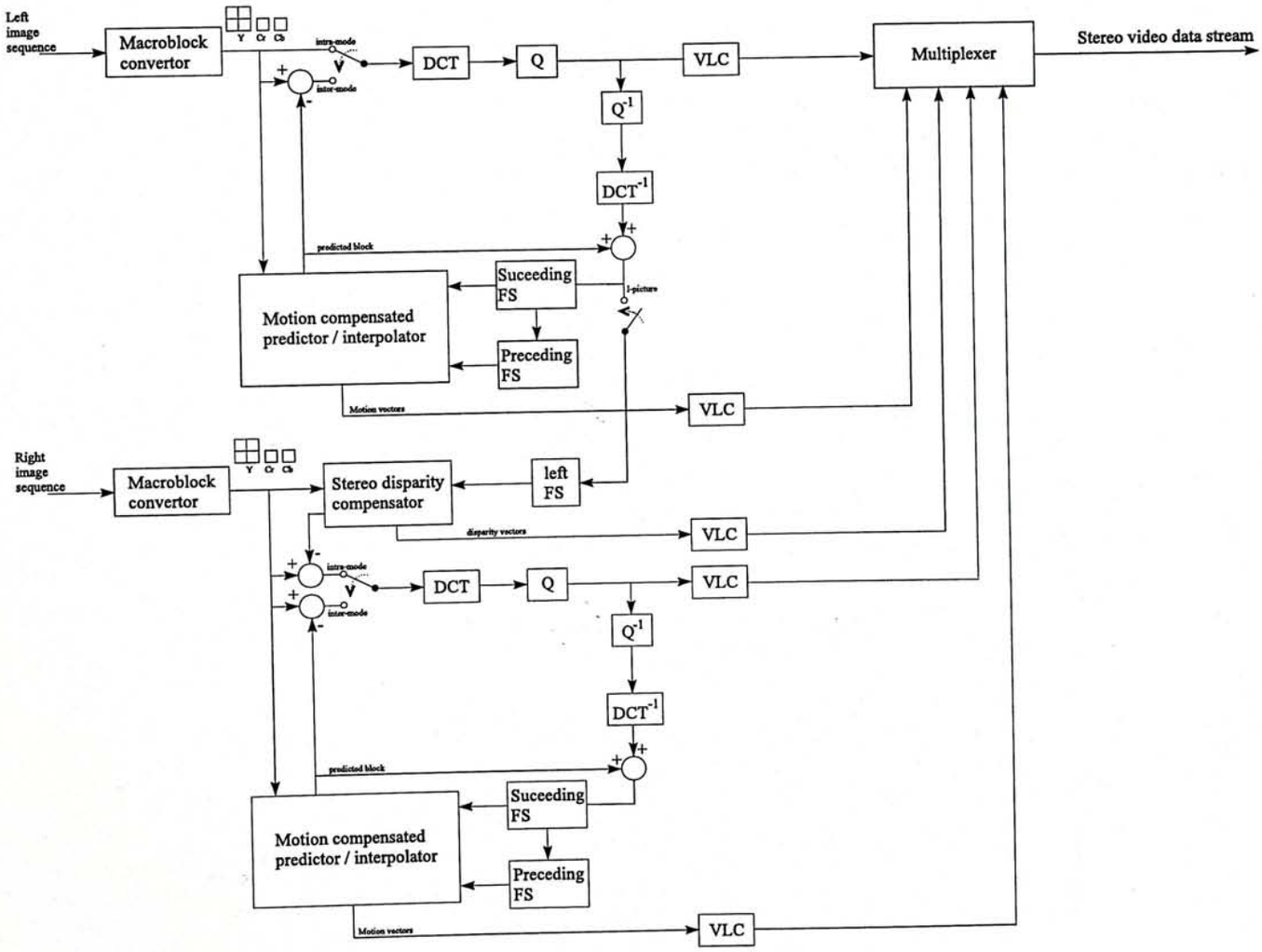


Figure 3.11: *Block operation of generic implementation.*

multiplexer. The main functional blocks will be discussed in the next subsections. The following description concentrates on the generation of stereo data stream by the generic design.

Each image frame in the left image sequence is first decomposed into a sequence of macroblocks by a macroblock converter. For each image frame coded to I-picture, all macroblocks in the frame are classified as intra-mode. The DCT coefficients of each 8×8 block are computed and each coefficient is quantized as described in appendix section A.4 for I-picture coding. The quantized DCT coefficients are then fed into two paths. One of them passes the quantized DCT coefficients into a variable length coder while the other path passes them into the decoding operations. The variable length coder identifies the quantized DC coefficients and the quantized AC coefficients and processing them accordingly. For the DC coefficients, DPCM is applied to the quantized data and the resulting difference data is coded using Huffman coding. For the AC coefficients, the quantized data are grouped into events, each consisting of a run of zeros followed by an amplitude and the events are coded using the hybrid Huffman/fixed length code as described in appendix section A.4.2.

In the decoding path, the coefficients are passed into an inverse quantizer and an inverse DCT operation to reconstruct the block being coded. For intra-mode blocks, there is no prediction block generated by the motion compensator and the reconstructed blocks are stored in a succeeding frame store directly. If the intra-mode blocks reconstructed belong to I-picture, the reconstructed blocks will also be fed into the left frame store for stereo disparity compensation. When all the reconstructed blocks for a complete image frame are ready, image frame stored in the succeeding frame store is a reconstructed image frame, which will be exactly the same at the one reconstructed at the decoder.

For P-picture coding, the image frame in the succeeding frame store is passed to the

preceding frame store. It is then used as an anchor by the motion compensator to perform forward prediction. The motion compensator decides whether the incoming block should be coded in intra-mode or inter-mode as mentioned in appendix section A.4 for picture coding of P-pictures. For intra-mode, the macroblock is processed the same way as the macroblocks for I-picture. For inter-mode, the predicted block generated by the motion compensator subtracted from the incoming block gives residual. This residual goes through the DCT operation and quantizer to reduce spatial redundancy. The quantized DCT coefficients for the residual are fed into the variable length coder as well as the decoding path. The variable length coder encodes the quantized coefficients the same as the coefficients for I-picture. The resultant VLC coded data as well as the VLC coded motion vectors generated by the motion compensator will be passed to the multiplexer. In the decoding path, image residual is reconstructed block by block through the inverse operators of quantizer and DCT. At this time, there is prediction block produced by the motion compensator. The predicted block and the reconstructed residual add up to produce a reconstructed block for the P-picture. It is then stored in the succeeding frame store. When all blocks for the P-picture are ready, image frame stored in the succeeding frame store is a reconstructed image frame for the P-picture.

For B-picture coding, the image frames in the succeeding and preceding frame stores are used as anchors by the motion compensator to perform bidirectional prediction. The rest of the operations is the same as for P-picture coding except that the motion vectors generated by the motion compensator consist of both forward and backward motion vector, and there is no frame store required for the storage of the reconstructed B-picture.

For the right image sequence, I-picture coding is different from I-picture coding of the left channel. The right image frame to be coded as I-picture is decomposed into macroblocks and then passed to the stereo disparity compensator. The stereo compensator performs stereo disparity estimation as described in section 3.4.1 with reference to the incoming

macroblocks and the reconstructed image frame stored in the left frame store. The disparity corrected prediction block so generated subtracted from the original right block gives the stereo residual. It is processed as intra-mode. The resultant VLC coded quantized DCT coefficients are passed to the multiplexer. The disparity vectors generated by the stereo compensator are also VLC coded and passed to the multiplexer. For the generation of P- and B-pictures of the right channel, exactly the same operations as those of the left channel is performed. All those parameters including quantized DCT coefficients for the residual and motion vectors are VLC coded and passed to the multiplexer. The multiplexer inserts all the codes into a bitstream at a suitable layer as shown in Figure 3.9. This generic scheme provides a more efficient generation of the stereo bitstream as no extra operations for extracting the I-pictures for the left channel and VLC decode are necessary before stereo disparity compensation.

3.5.1 Macroblock Converter

The macroblock converters are color space converters in which the image frame is converted to a color space with separate luminance and chrominance components. This is done because the human eye is far more sensitive to the luminance information (Y) than it is to the chrominance information (Cr and Cb); by separating them, it is possible to compress the chrominance information more than the luminance without the perceived image quality suffering. The macroblock converters first decompose each incoming image frame into 16×16 pixel sections, perform the R-G-B to Y-Cr-Cb conversion to each pixel in the sections, and subdivide the resultant Y-Cr-Cb data into six 8×8 blocks. The R-G-B to Y-Cr-Cb conversion is defined as

$$\begin{pmatrix} Y \\ C_r \\ C_b \end{pmatrix} = \begin{pmatrix} 0.2990 & 0.5870 & 0.1140 \\ -0.1687 & -0.3313 & 0.5000 \\ 0.5000 & -0.4187 & -0.0813 \end{pmatrix} \begin{pmatrix} R \\ G \\ B \end{pmatrix} \tag{3.5}$$

Since the conversion is a one-to-one mapping, the resultant image data consist of 16×16 YCrCb triplets. Chrominance subsampling is applied to the Cr and Cb values by taking average of the four nearby pixels as described in appendix section A.2 so that 8×8

blocks of Cr and Cb result. In addition, the 16×16 Y component is subdivied into four blocks of 8×8 luminance values. These six blocks form a macroblock and pass for further operations.

3.5.2 DCT Functional Block

This is the main functional block for all MPEG encoder and decoder. The implementation of the two-dimensional DCT for each 8×8 block was based on double precision data type, i.e., 64 bits floating points representation, operated on equations (2.13) and (2.14). Because the 2-D DCT is separable, the summations was done as eight 1-D DCTs on all rows followed by eight 1-D DCTs on the eight columns. The equations are rearranged as follows:

FDCT:

$$v(k,l) = \sum_{m=0}^{7} \alpha(k) \cos\left[\frac{(2m+1)k\pi}{16}\right] \sum_{n=0}^{7} \alpha(l) u(m,n) \cos\left[\frac{(2n+1)l\pi}{16}\right], \qquad 0 \le k,l \le 7 \tag{3.6}$$

IDCT:

$$u(m,n) = \sum_{k=0}^{7} \alpha(k) \cos\left[\frac{(2m+1)k\pi}{16}\right] \sum_{l=0}^{7} \alpha(l) v(k,l) \cos\left[\frac{(2n+1)l\pi}{16}\right], \qquad 0 \le m,n \le 7. \tag{3.7}$$

where

$$\alpha(k) = \begin{cases} \sqrt{\frac{1}{8}} & \text{for } k = 0 \\ \frac{1}{2} & \text{for } k = 1,2,\ldots,7. \end{cases} \tag{3.8}$$

A faster DCT implementation according to Chen *et al.* [47] were implemented. The algorithm exploits the symmetries of the DCT. Considering the following 1-D DCT equation,

$$v(l) = \sum_{n=0}^{7} \alpha(l) u(n) \cos\left[\frac{(2n+1)l\pi}{16}\right], \qquad 0 \le l \le 7.$$

For the case $l = 2$,

$$u(2) = \frac{u(0)}{2} \cos\left[\frac{2\pi}{16}\right] + \frac{u(1)}{2} \cos\left[\frac{6\pi}{16}\right] + \frac{u(2)}{2} \cos\left[\frac{10\pi}{16}\right] + \frac{u(3)}{2} \cos\left[\frac{14\pi}{16}\right] +$$

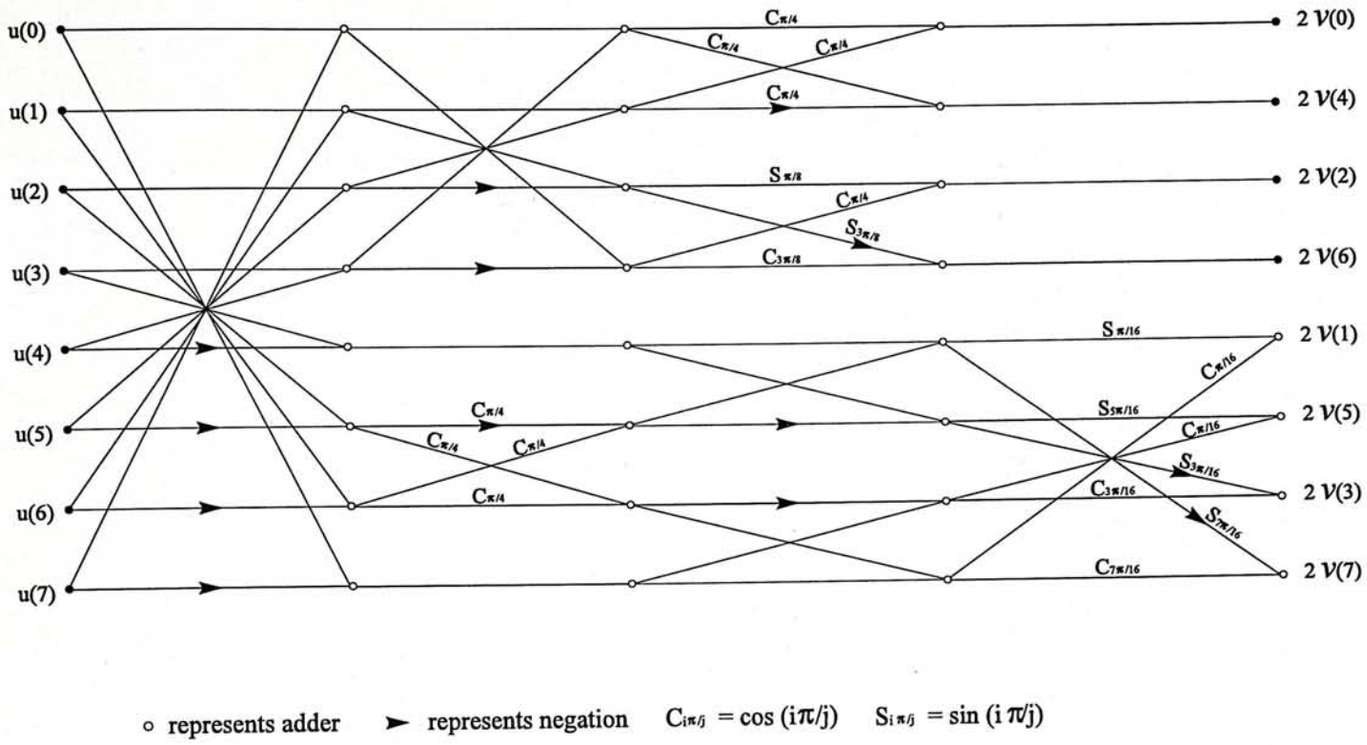


Figure 3.12: *Signal flowgraph for* 8×8 *fast DCT.*

$$\begin{aligned} & \frac{u(4)}{2}\cos\left[\frac{18\pi}{16}\right] + \frac{u(5)}{2}\cos\left[\frac{22\pi}{16}\right] + \frac{u(6)}{2}\cos\left[\frac{26\pi}{16}\right] + \frac{u(7)}{2}\cos\left[\frac{30\pi}{16}\right] \\ = \; & \frac{u(0)}{2}\cos\left[\frac{\pi}{8}\right] + \frac{u(1)}{2}\sin\left[\frac{\pi}{8}\right] - \frac{u(2)}{2}\sin\left[\frac{\pi}{8}\right] - \frac{u(3)}{2}\cos\left[\frac{\pi}{8}\right] - \\ & \frac{u(4)}{2}\cos\left[\frac{\pi}{8}\right] - \frac{u(5)}{2}\sin\left[\frac{\pi}{8}\right] + \frac{u(6)}{2}\sin\left[\frac{\pi}{8}\right] + \frac{u(7)}{2}\cos\left[\frac{\pi}{8}\right] \end{aligned}$$

where the equality $\cos\left[\frac{k\pi}{8}\right] = \sin\left[\frac{\pi}{2} - \frac{k\pi}{8}\right]$ is used. For the rest of $v(l)$, the similar equations can be constructed. The equations are constructed to minimize the number of multiplications. The signal flowgraph for the fast DCT algorithm is shown in Figure 3.12. In the figure, the flow of operations is from left to right, and lines are summed where they merge at a node. If a line contains an arrow, the signal is negated before the addition to (i.e., subtracted from) the other signal fed to the node. Multiplication of a signal is indicated above a line. The corresponding flowgraph for the inverse DCT can be obtained by reversing the direction of the signals. This is valid because the DCT is an orthogonal transform. This algorithm involves 26 real additions and 16 real multiplications per 1-D DCT operation. For 2-D DCT, only 52 real additions and 32 real multiplications are required.

3.5.3 Rate Control

As the coding algorithm used by MPEG is variable bit rate, rate regulation buffer is necessary to produce constant rate for transmission. Besides, it is much easier to vary to the number of bits allocated to the right channel through this mechanism in order to study the characteristic of the stereo encoder. The principle of the rate control mechanism is based on the facts that lower bit rate is obtained for coarser quantization step size and higher bit rate is obtained for finer quantization step size. Accordingly, MPEG defines a parameter called *quantizer_scale*, Q_p, which is a scaling factor to quantization matrix. Q_p has a linear relationship to the output buffer fullness and it is updated whenever the buffer fullness is updated.

The evaluation of the buffer fullness is done by first assigning initial bits per update interval to each picture type in a GOP. Let N_i, N_p and N_b be the assigned bits for I-picture, P-picture and B-picture, respectively, in a GOP. Then, the frame rate for each picture type can be calculated from the preset bit rate. Suppose I_s, P_s and B_s be the corresponding frame rates obtained for I-picture, P-picture and B-picture. Desired bit rate is given by

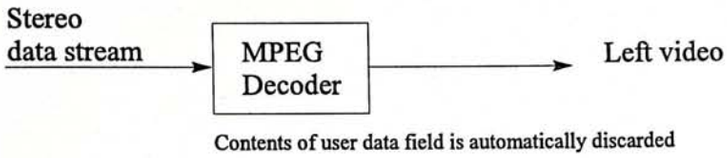
$$R_d = N(I_s N_i + P_s N_p + B_s N_b) \tag{3.9}$$

where N is the number of pictures per update interval. At update, the number of bits generated during last interval is added up while N_i, N_p or N_b is subtracted according to picture type. In addition, calculated bit rate, R_c, is obtained based on actual number of bits consumed. Finally, target bit rates are adjusted by $\frac{R_d}{R_c}$.

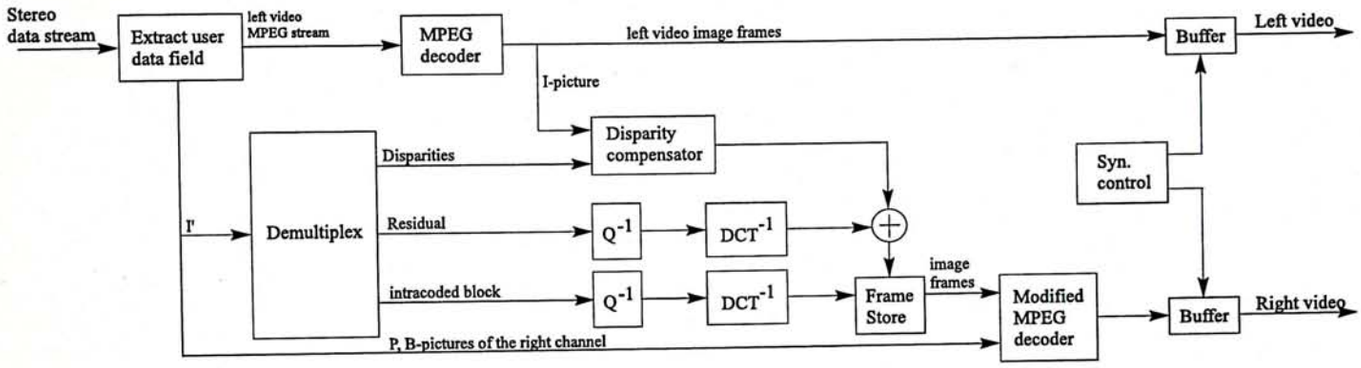
For each macroblock coding, the assignment of Q_p is as follows.

$$Q_p = \left[d + \text{actual bits} - S_i - k \times \left(\frac{T}{\text{Total number of macroblocks in a picture}} \right) \right] \times \frac{31}{r} \tag{3.10}$$

where d is the buffer fullness; S_i is the number of bits used for side information (i.e. bits not used for DCT coefficients); k represents the number of macroblocks being coded; T



Mono playback



Stereo playback

Figure 3.13: *Stereoscopic video decoder.*

is the target bit rate; r is the reaction parameter and is given by

$$r = 2 \times \frac{\text{bit rate}}{\text{frame rate}}. \tag{3.11}$$

3.6 Stereoscopic MPEG Decoder

Stereoscopic decoder is the inverse operation of the encoder. It is considerably simpler than encoding since it is not necessary to include motion estimation and stereo disparity estimation. Figure 3.13 shows the decoding of the stereo MPEG bitstream. The following description will discuss mono playback and stereo playback separately.

3.6.1 Mono Playback

When the stereo data stream is input to a normal MPEG decoder, the decoder reads the data stream and decodes the data elements in the stream according to the defined syntax. The information inserted in the user data fields is discarded at this stage and

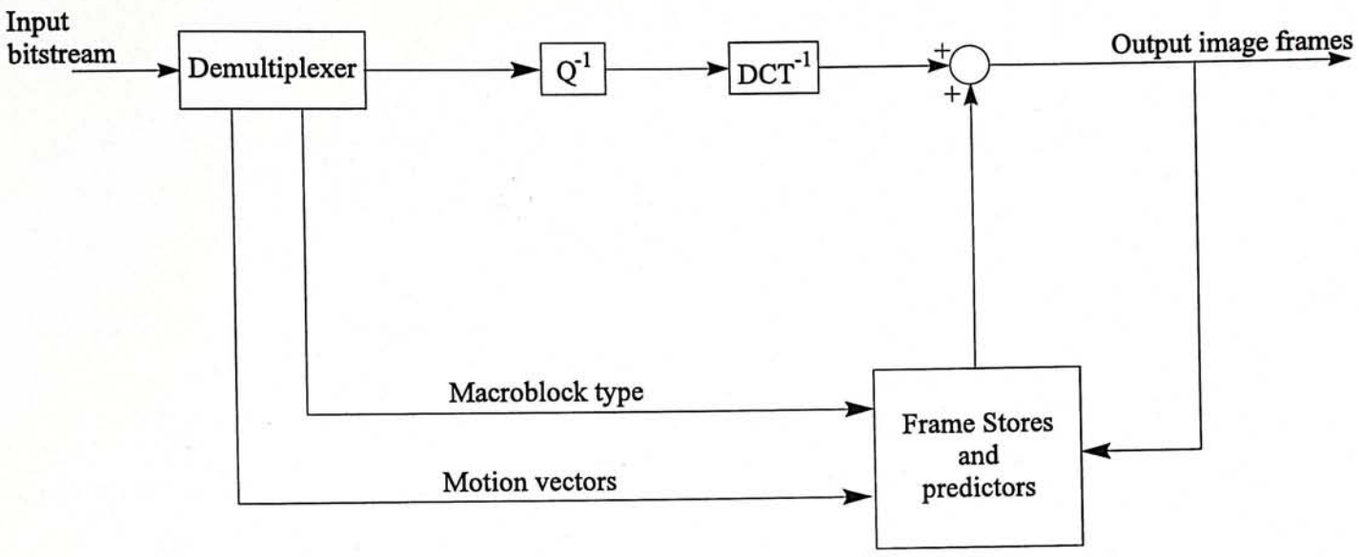


Figure 3.14: *MPEG decoder.*

therefore only data elements of the left channel retain for further operation. Figure 3.14 shows the main functional blocks of an MPEG decoder.

As the decoder reads the stream, it identifies the start of a coded picture and then the type of the picture. It demultiplexs the macroblock type and the motion vectors if the picture being decoded is either P- or B-picture according to the coding syntax. For P- and B-pictures, the motion vectors are used to construct a prediction of the current macroblock based on the preceding and succeeding image frames stored in the frame stores. The VLC coded DCT coefficients are decoded and inverse quantized. The resultant data are then transformed by an inverse DCT operator, and the inverse transformed data is added to the predicted macroblock for P- or B-picture. After all the macroblocks in the picture have been processed, the image frame is completely reconstructed.

The image frame decoded from I- or P-picture is stored in the frame store and is used by the predictor as reference frame for subsequent pictures decoding. Before image frames are displayed they must be re-ordered from the coding order to their temporal order. After re-ordering, the image frames are available in YCrCb format, they may need to convert to RGB format by post-processing. This conversion can be described by the

following matrix equation.

$$\begin{pmatrix} R \\ G \\ B \end{pmatrix} = \begin{pmatrix} 1 & 0 & 1.40200 \\ 1 & -0.34414 & -0.71414 \\ 1 & 1.77200 & 0 \end{pmatrix} \begin{pmatrix} Y \\ C_r \\ C_b \end{pmatrix} \tag{3.12}$$

3.6.2 Stereo Playback

A stereo decoder extracts the stereo information from the user data fields in the stereo data stream before passing the remaining bitstream to a normal MPEG decoder to generate the left image sequence. If the picture decoded is an I-picture, the reconstructed left image frame is passed to a disparity compensator for further operation. From the extracted data, the coded pictures, I'-pictures, of the right channel corresponding to I-pictures of the left channel are identified. They are output to a demultiplexer in which the stereo disparity vectors, DCT coefficients of the stereo residual and DCT coefficients of the intracoded blocks are extracted. The disparity compensator uses the stereo disparity vectors and the reconstructed I-picture of the left channel to predict the disparity corrected macroblocks of the right channel. The DCT coefficients of the stereo residual and intracoded blocks are inverse quantized. Each 8×8 block of the coefficient data in a macroblock is transformed by an inverse DCT operator. The resultant stereo residual which is occluded area for the left view is added to the disparity corrected macroblocks and is stored in a frame store while the resultant intrablock is stored in the same frame store directly. After all the macroblocks in the I'-picture have been processed the image frame stored is completely reconstructed and it is passed to a modified MPEG decoder for further operation.

The modified MPEG decoder shown in Figure 3.15 is very similar to the original MPEG decoder except that it is not necessary to decode the macroblocks of I-pictures. The reconstructed image frames decoded from I'-pictures are stored in the frame stores directly. The demultiplexer decodes VLC coded data of the bitstream to produce the quantized DCT coefficients of the P-, B-pictures. These are assembled for each 8×8 block of pixels

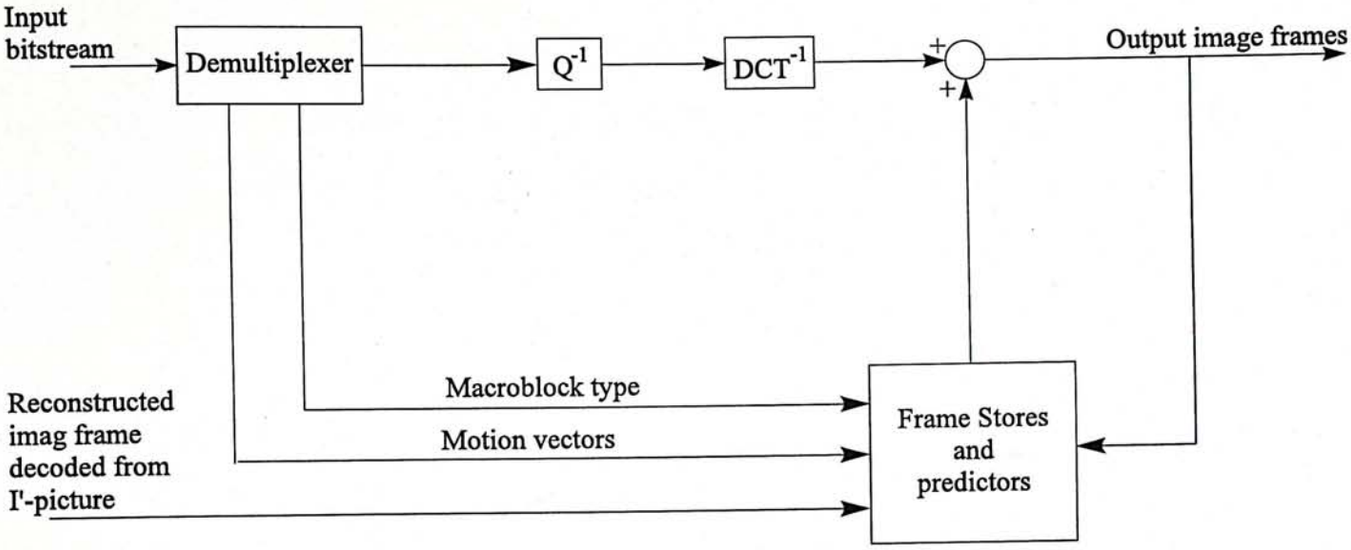


Figure 3.15: *Modified MPEG decoder.*

in the corresponding image frame. The inverse quantizer produces the actual DCT coefficients. The resultant coefficients are then transformed into pixel values, which are the temporal residual, by the inverse DCT transformer. If the residual belongs to P-picture, there are motion vectors decoded by the demultiplexer passing to the predictor. Having got these motion vectors, the predictor applys them to the blocks in the stored image frame (decoded from I'-picture) to produce the predicted blocks. The residual is added to the prediction block by block. The resultant block is stored in the frame store and output to the right channel buffer shown in Figure 3.13.

The frame stores provide two storage spaces, one for preceding frame and the other for succeeding frame. For the decoding of B-pictures, both frame stores are used for bidirectional prediction. The residual for B-pictures is decoded by the same operations as described for P-pictures. As the data for I'-, P- and B-pictures are extracted sequentially, the reconstructed image frame from I'-picture will not come at the same moment with the one from P-picture. Furthermore, the reconstructed image frame from the P-picture is put into the succeeding frame after the succeeding frame store passes its contents to the preceding frame store. There is no data congestion at the frame stores. The decoded image frame will be output one by one.

Referring back to Figure 3.13, when both the image buffers get the image frames for the left and right channels respectively, they are re-ordered from the coding order to their temporal order. After that synchronization controller triggers the buffers to output the image frames for display.

Chapter 4

Performance Evaluation

4.1 Introduction

This chapter describes the tests and presents the experimental results on the performance of the proposed stereoscopic video coding scheme. A set of stereo motion video clips were computer generated for the use in the testings. The following section describes the generation of the video test sequences. Section 4.3 presents test platform used for the testings. Section 4.4 presents the simulation results with objective measurement of the decoded image qualities and subjective visual quality evaluation of the scheme.

4.2 Test Sequences Generation

To evaluate the performance of any video coding algorithm, test video sequences are required. The Simulation Model Editorial Group of ISO-IEC working group issued the MPEG Video Simulation Model Three (SM3) report [48], in which video test sequences are defined for the evaluation of the MPEG source coder. However, there was no test video sequence defined for the evaluation of the stereo coding algorithm. In view of that, we took our way to create a set of stereo video test sequences. Two computer software applications, Autodesk 3-D Studio published by Yost Group Inc. and Vistapro published by Virtual Reality Laboratories Inc. were considered to be used for the production

of the test stereoscopic motion video. As Vistapro is a 3-D landscape generator, it produces stereoscopic image sequence by moving a pair of cameras in a scene of fractal landscape. Image sequences are rendered frame by frame. Autodesk 3-D Studio produces stereoscopic image sequence in a different way. Objects with motion defined are created and placed in a scene. Two camera objects having slightly shifted position are defined for capturing the view sequences. Although camera motions can also be defined, it will complicate specification of the scene a lot. Thus, stationary cameras as if a man watching moving objects are used. As the motion videos generated by Autodesk 3-D Studio resembles closer to the scene in real world, it was selected to generate test sequences. The test Image sequences were generated frame by frame in Targa file format, or TGA, which supports images of any color depth 1 and 32 bits.

Four sets of stereoscopic 40 frames video sequences were generated. They are 24-bit color with a frame resolution of 512×480 pixels, which is similar to NTSC broadcast quality. The simplest one is a scene with a dark background containing a burning candle and ball bouncing vertically (Figure 4.4). The second video sequence is a cactus placed in front of a window, which contains more motion objects compared to the previous sequences. The cactus stretches its arms and legs and then stands up (Figure 4.3). The other two video sequences are the most complex ones. They contain a dragon who moves its head to the cameras and open its mouth providing the strongest depth perception. These two video sequences were generated from the same scene. One of them contains fast motion of the dragon (Figure 4.5) but the other only contains slower motion with the close-up view of the dragon's head (Figure 4.6).

4.3 Simulation Environment

The coding algorithms were developed on a PC based platform. All the video sequences were initially generated in TGA format and subsequently converted into YCrCb format. All the program modules developed for the coding algorithms operate on the YCrCb

files.

The stereoscopic coding algorithms were coded in C-language, based on the source code of an existing MPEG encoder. For monocular viewing, an MPEG player for Windows 3.1 called VMPEG developed by Stefan Eckart was used. VMPEG can decode the stereoscopic video file with left channel displayed on the screen.

4.4 Computer Simulation

The computer generated test sequences mentioned in section 4.2 were used to test the algorithm. The number of frames contained in a group of pictures was set to 6, which included 1× I-picture, 2× P-pictures and 3× B-pictures. As the simulation was to investigate the objective performance of the coding scheme with different compression of the right channel, rate control strategy is applied. In the previous chapter, it is described that there are three picture types: I'-picture, P-picture, and B-picture for the right channel. Both I'-picture and P-picture are predictive pictures. I'-picture is predicted with reference to the corresponding left image frame while P-picture is predicted with reference to the preceding image frame which may be a decoded I'-picture or a decoded P-picture. They are assigned the same number of bits initially. B-picture is assigned the smallest number of bits since it will not be used as reference and thus does not propagate error into other pictures. The rest of the rate control mechanism is as mentioned in section 3.5.3.

4.4.1 Objective Results

An objective measurement of the decompressed image qualities was performed by comparing the peak signal-to-noise ratio (or *PSNR*). Because these simulations were done by decompressing the images to three channel color images (Y, Cr and Cb), the *PSNR* results reported is in terms of an average *PSNR* defined over the Y, Cr, and Cb components. The contribution of the luminance component for a macroblock is $N \times N$, and the

contribution of the two chrominance components for a macroblock is $\frac{N}{2} \times \frac{N}{2}$; therefore, the total number of pixels is $1.5N^2$, where $N = 16$. Thus the *PSNR* is defined as

$$PSNR = 10 \log_{10} \frac{1.5N^2 255^2}{\sum_{i=1}^{3} \sum_{m=1}^{k} \sum_{n=1}^{k} [x_i(m,n) - \hat{x}_i(m,n)]^2}, \tag{4.1}$$

where

$$k = \begin{cases} N & \text{for } i = 1 \\ \frac{N}{2} & \text{for } i = 2, 3 \end{cases} \tag{4.2}$$

; $x_i(m,n)$ and $\hat{x}_i(m,n)$ are the original and the reconstructed value of a pixel at row m and column n for components i, respectively.

In Figure 4.1, the *PSNR*'s of the right channel over 30 frames of the sequence "cactus" and "candle and ball" are shown, where the graphs for five different bit rates varying from 100% to 60% bit rate of the left channel with 10% step size are presented to show the variation of the *PSNR* versus frame number at those bit rates. The graphs show the same pattern for different bit rates and the behavior of the *PSNR*'s show some regular periodicity inside each GOP. The first frame for each GOP is reconstructed from disparity prediction. In the "candle and ball" sequence, the disparity predicted frames give higher *PSNR* than the following frame, which is reconstructed from B-picture while the disparity predicted frames generated from "cactus" sequence give lower *PSNR*. It is because the separation between the cameras for capturing the "cactus" sequence is greater than that for capturing the "candle and ball" sequence, which is intended to do so to exaggerate the stereoscopic effect; hence the frames in the "cactus" sequence contains greater disparities. On the other hand, in the "candle and ball" sequence the motion of the bouncing ball is very fast; therefore the interframe residual consumes more bits. Conversely, the "cactus" sequence contains less motion; hence *PSNR*'s obtained from the interframe prediction are higher than those obtained from the I'-pictures. In addition, it can be seen that the *PSNR*'s for the "cactus" sequence gives more regular pattern. This is also ascribable to the non-constant motion of the bouncing ball due to gravity; thus the lower the ball, the faster the motion and vice versa. It is also observed from the graph that the magnitude

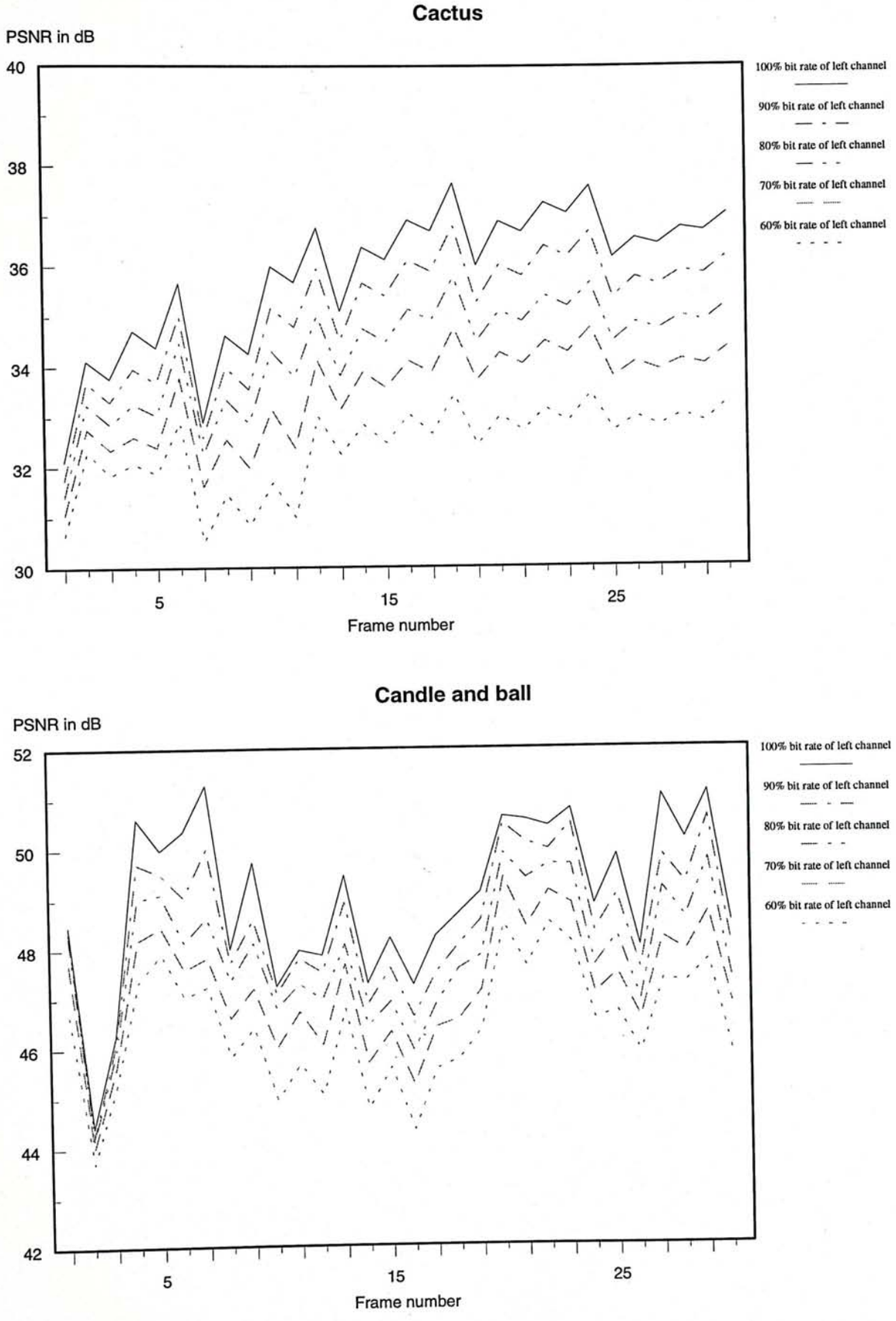


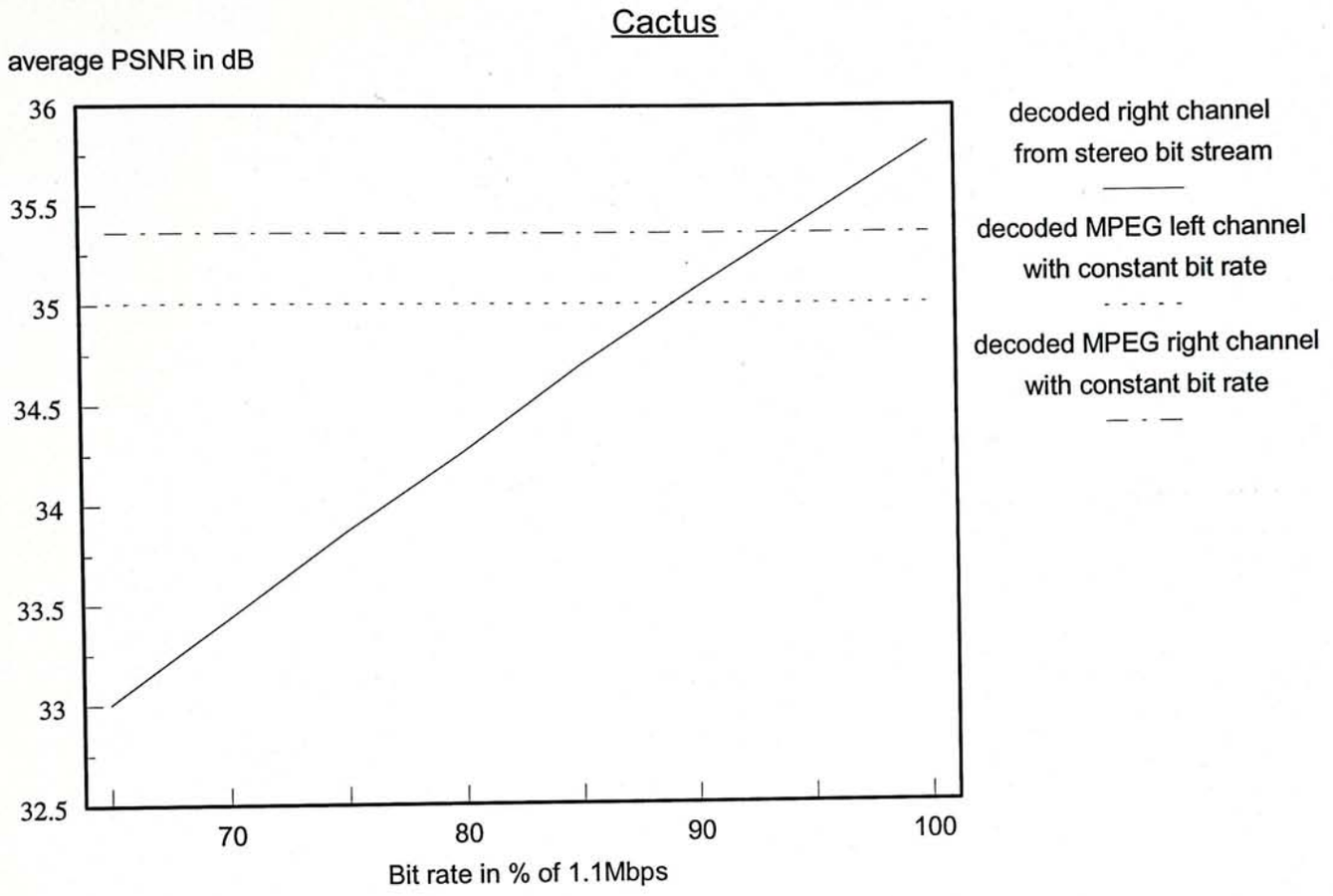
Figure 4.1: *Variation of PSNR of the right channel versus frame number for different bit rates.*

of *PSNR* depends on the complexity of the scenes. For simple scene such as the "candle and ball" higher *PSNR* was produced. Comparing the two graphs, difference over 10 dB is obtained.

In order to investigate the relationship between *PSNR* and compression ratio, the target bit rate of the additional right channel was adjusted by multiplying a factor to 1.1 Mbps, that is the target bit rate for normal MPEG video applications, while the bit rate for the left channel was unaffected and was set to 1.1 Mbps. Ideally, we would like to obtain lowest possible bit rate with highest possible *PSNR*. Figure 4.2 shows average *PSNR* against bit rate of the right channel for the four video sequences: "cactus", "candle and ball", "dragon I", and "dragon II". The horizontal dotted line is the average *PSNR* of the left channel video sequence which was kept constant in this evaluation. The horizontal dash-dot-dash line is the average *PSNR* of the right channel video sequence decoded from its normal MPEG coded bitstream at 1.1 Mbps. It is plotted for comparing the performance of the proposed scheme with the normal MPEG encoder.

Generally, the *PSNR*'s of the left and right parts of a stereoscopic video sequence are different even if the same MPEG encoder at the same bit rate is used. It is due to the two views containing occluded regions which are different and lead to different decompressed qualities. For the average *PSNR*'s of the left and right channels reaching 35 dB, the different *PSNR*'s are less than 1 dB, which is only 28%. By comparing the average *PSNR* of the right channel decoded from normal MPEG bitstream (the dash-dot-dash line) with the one decoded from the stereoscopic data stream (the solid line). It is found that at the same bit rate (1.1 Mbps), the average *PSNR* generated from the proposed scheme is always higher than the one generated from normal MPEG encoder.

- For the "cactus" sequence, when the bit rate drops to around 96% of 1.1 Mbps, comparable MPEG decoded *PSNR* of the right channel is obtained. Thus, 2% further compression is achieved comparing to two MPEG video sequences. For the same left and right *PSNR*'s, the bit rate is allowed to drop to around 90% of 1.1



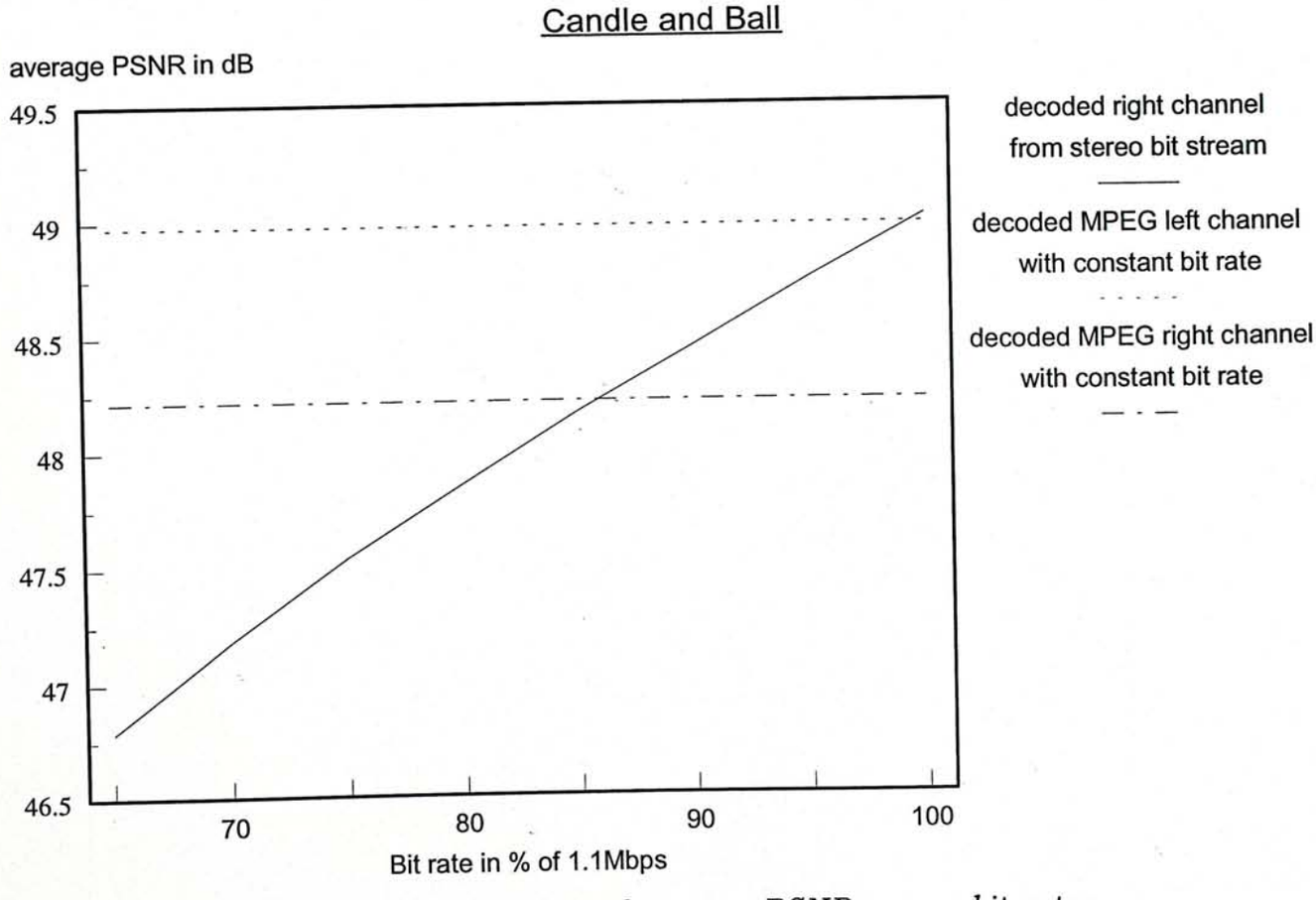
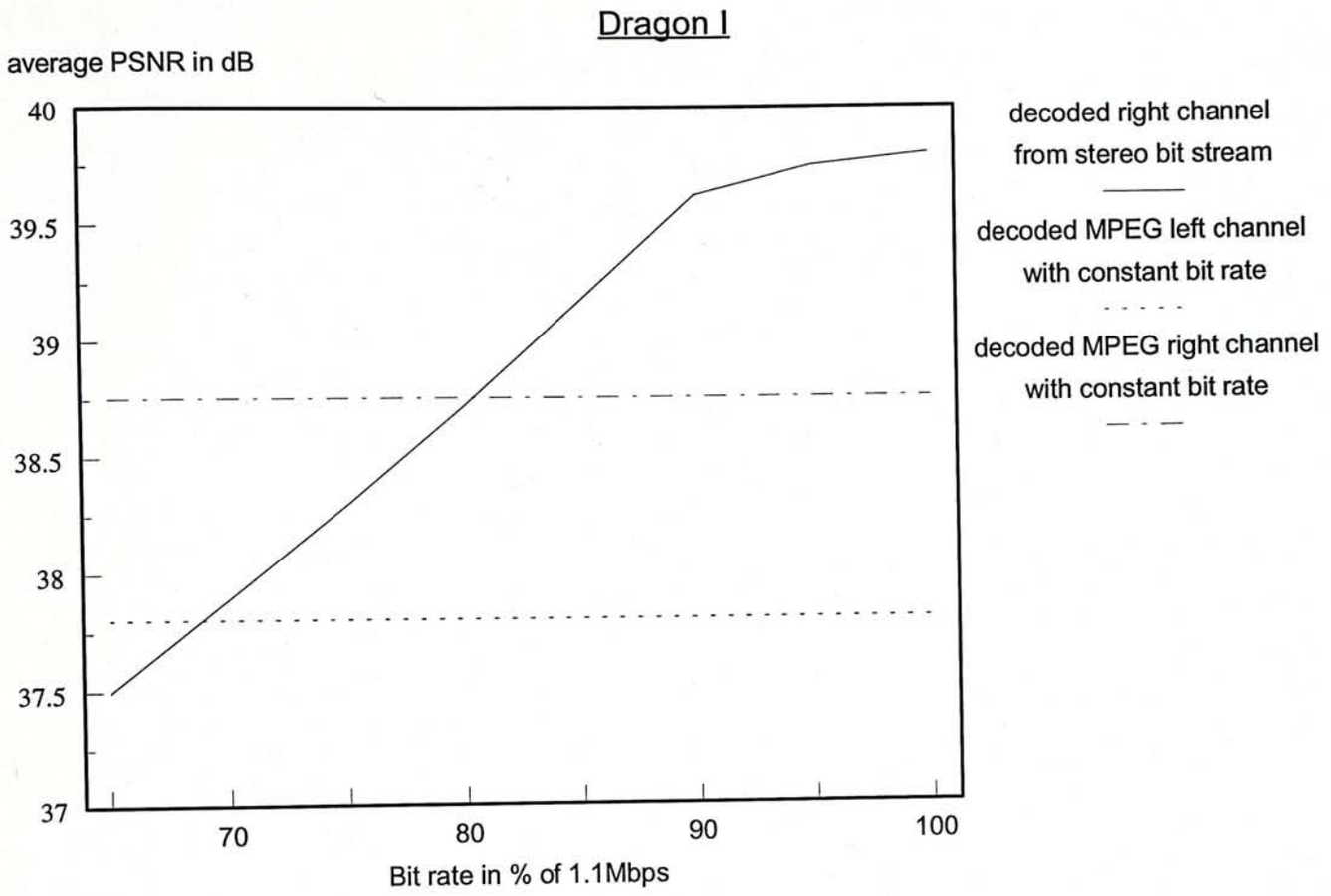


Figure 4.2a: *Variation of average PSNR versus bit rates.*



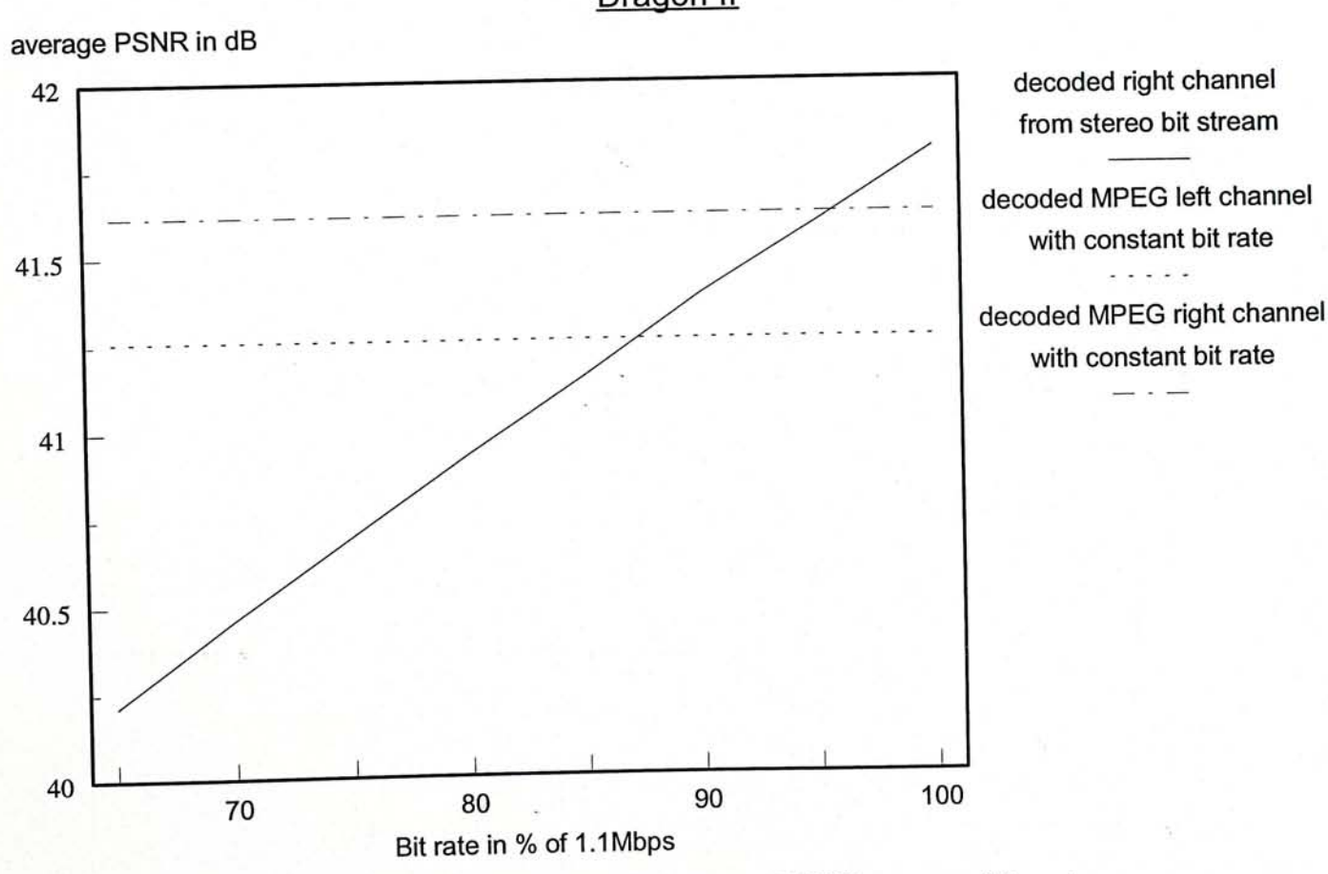


Figure 4.2b: *Variation of average PSNR versus bit rates.*

Mbps, which gives 5% further compression.

- For the "candle and ball" sequence, when the bit rate drops to around 86% of 1.1 Mbps, comparable MPEG decoded *PSNR* of the right channel is obtained. In this case, 7% further compression is achieved. It is observed that higher *PSNR*'s comparing with the "cactus" sequence is obtained and this is the result of taking the average of the *PSNR*'s for different frame number shown in Figure 4.1. As the average *PSNR* is affected by the *PSNR*'s of the disparity predicted frames, the greater separation between the cameras for capturing the "cactus" sequence also accounts for the lower average *PSNR* of the "cactus" sequence. Besides, the simple scene of "candle and ball" is the other reason. As its background is dark, there is few AC coefficients as well as DC coefficients after DCT transform. The zero run-length coding generates nearly zero overhead for those blocks in the background. Therefore, more bits are provided for the coding of the intra-coded blocks for the foreground; hence higher *PSNR*'s.

- For the "dragon I" sequence, when the bit rate drops to around 81% of 1.1 Mbps, comparable MPEG decoded *PSNR* of the right channel is obtained. This gives 4.5% further compression comparing to two MPEG video sequences. For the same left and right *PSNR*'s, the bit rate is allowed to drop to around 70% of 1.1 Mbps, which gives 15% further compression. It is also shown that the *PSNR* of the reproduced right channel image sequence increases less rapidly at bit rate above 90% of 1.1 Mbps. The change of the increasing rate of the *PSNR* is mainly because the "dragon I" sequence varies from wide view of the scene, where the complete dragon was captured, to close-up view of the dragon, where only the dragon's head was captured. The interframe residual for both channel contains a lot of image contents which are coded as intra-coded blocks. When the right channel bit rate increases to above 90% of 1.1 Mbps, the *PSNR* of the decompressed intra-coded blocks becomes stable and is less affected by the increasing number of bits of the coded DCT coefficients.

- For the "dragon II" sequence, when the bit rate drops to around 96% of 1.1 Mbps, comparable MPEG decoded *PSNR* of the right channel is obtained. Thus, 2% further compression is achieved comparing to two MPEG coded video sequences. For the same left and right *PSNR*'s, the bit rate is allowed to drop to around 87% of 1.1 Mbps, which gives 6.5% further compression. This sequence is captured from the same scene as "dragon I" but with different location of the cameras and with slower motion. Thus, the interframe residual is less comparing with "dragon I" sequence. In addition, this sequence only contains the closed-up view. This benefits the disparity prediction; hence higher *PSNR*'s are results.

After analysing the dependency of the *PSNR* and bit rate of the right channel video, it can be concluded that the objective coding quality measure depends not only on the complexity of the scenes but also the relative motion of the objects in the scenes.

4.4.2 Subjective Results

For a viewer the subjective visual quality of the reconstructed video is an important issue. In our subjective tests, video sequences were reconstructed and displayed on screen for visual comparision. As we did not have device to view the decompressed stereoscopic image sequence. The left and right image frames were displayed simultaneously and compared visually. It was found that although the *PSNR*'s between the left and right image of the decompressed MPEG coded sequences were different (as shown in Figure 4.2, the greatest difference of the two horizontal lines is 0.9dB), they have comparable subjective qualities. Thus, we concluded that for less than 1 dB *PSNR* differences, nearly the same subjective qualities were observed.

In order to perform subjective comparison with the normal MPEG coded right channel, image frames decoded from normal MPEG coded bitstream were displayed on screen while one of the corresponding decoded image frame from stereoscopic data stream with different bit rates was chosen randomly and displayed simultaneously on screen. Five

people were told to find the image frame with better quality. It was found that 70% of 1.1 Mbps for the right channel shows nearly the same visual quality with the normal MPEG decoded right image frame. This gave compression ratio about 15% comparing to two MPEG coded video sequences. Although bit rate below 70% of 1.1 Mbps results in observable visual degrade, it is believed that the depth perception will still remain if the sequences are displayed at video rate, i.e., higher than 25 frames/sec.

Some sample image pairs are shown in Figure 4.3 to 4.6 where three selected image pairs are demonstrated. For the figures, the top pair indicates the image frames corresponding to frame number 12. They are decompressed from intra-coded picture for the left image frame and disparity compensated picture for the right image frame. The middle pair indicates the image frames corresponding to frame number 26. They are decompressed from predicted pictures. The lower pair indicates the image frames corresponding to frame number 37. They are decompressed from interpolated pictures. Figure 4.7 and Figure 4.8 show a selected subjective test images for "cactus" and "dragon I", respectively. The upper image pair is decoded from normal MPEG coded left and right video sequence while the lower eight image frames shows the gradually degraded qualities of the corresponding image frames decoded from stereo data stream.

I-picture of the left channel
(frame number = 12)

I'-picture of the right channel
(frame number = 12)

I-picture of the left channel
(frame number = 26)

P-picture of the right channel
(frame number = 26)

B-picture of the left channel
(frame number = 37)

B-picture of the right channel
(frame number = 37)

Figure 4.3: *Selected image pairs for "cactus" sequence.*

I-picture of left channel
(frame number = 12)

I'-picture of right channel
(frame number = 12)

P-picture of left channel
(frame number = 26)

P-picture of right channel
(frame number = 26)

B-picture of left channel
(frame number = 37)

B-picture of right channel
(frame number = 37)

Figure 4.4: *Selected image pairs for "candle and ball" sequence.*

I-picture of left channel
(frame number = 12)

I'-picture of right channel
(frame number = 12)

P-picture of left channel
(frame number = 26)

P-picture of right channel
(frame number = 26)

B-picture of left channel
(frame number = 37)

B-picture of right channel
(frame number = 37)

Figure 4.5: *Selected image pairs for "dragon I" sequence.*

I-picture of left channel
(frame number = 12)

I'-picture of right channel
(frame number = 12)

P-picture of left channel
(frame number = 26)

P-picture of right channel
(frame number = 26)

B-picture of left channel
(frame number = 37)

B-picture of right channel
(frame number = 37)

Figure 4.6: *Selected image pairs for "dragon II" sequence.*

MPEG decoded left image
at 1.1Mbps

MPEG decoded right image
at 1.1Mbps

Stereo decoded right image
at 100% of 1.1Mbps

Stereo decoded right image
at 95% of 1.1Mbps

Stereo decoded right image
at 90% of 1.1Mbps

Stereo decoded right image
at 85% of 1.1Mbps

Stereo decoded right image
at 75% of 1.1Mbps

Stereo decoded right image
at 70% of 1.1Mbps

Stereo decoded right image
at 65% of 1.1Mbps

Stereo decoded right image
at 60% of 1.1Mbps

Figure 4.7: *Subjective test for "Cactus".*

MPEG decoded left image
at 1.1Mbps

MPEG decoded right image
at 1.1Mbps

Stereo decoded right image
at 100% of 1.1Mbps

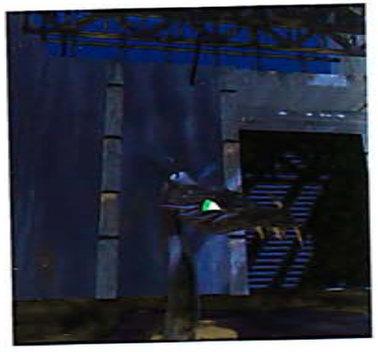
Stereo decoded right image
at 95% of 1.1Mbps

Stereo decoded right image
at 90% of 1.1Mbps

Stereo decoded right image
at 85% of 1.1Mbps

Stereo decoded right image
at 75% of 1.1Mbps

Stereo decoded right image
at 70% of 1.1Mbps



Stereo decoded right image
at 65% of 1.1Mbps

Stereo decoded right image
at 60% of 1.1Mbps

Figure 4.8: *Subjective test for "Dragon I".*

Chapter 5

Conclusions

In this study, a stereoscopic video coding scheme has been designed for the efficient encoding of two channel stereo motion video with the resultant code playable in a non-stereo MPEG digital video system. The scheme based on the fact that stereo motion video consists of two views of the same scence from a slightly displaced view points. Therefore the views are similar with some disparities (stereoscopic information). By using the left video channel as reference and finding out the stereoscopic information against the right video channel so that the right channel can be reconstructed from the reference channel using the stereoscopic information, the right channel becomes redundant. In order to maintain compatibility with the MPEG video coding standard, the reference left channel of video is encoded in MPEG format with the stereoscopic information inserted at fields reserved for user data. Since MPEG is already a highly efficient coding method, the compression ratio depends on the size of the stereoscopic information.

Two approaches had been considered to extract the stereoscopic information: coding by stereoscopic differences and I-picture only disparity coding. In the first approach, a sequence of stereoscopic differences was obtained by subtracting the left video image sequence with the right sequence frame by frame. Then, two methods were attempted to further process the resultant sequence. In the first one, the stereoscopic difference sequence was coded by Discrete Cosine Transform (DCT) frame by frame. As this method

only deals with the spatial redundancy within the stereoscopic differences, the resultant data is too large to be practical. The second method exploits both the spatial and temporal redundancies within the stereoscopic differences. However, it was found that this method still could not reduce the size to an usable value. Consequently, a different approach was attempted.

In the I-picture only disparity coding, disparity compensation was utilized to exploit the redundancy between the stereo image pairs corresponding to the I-pictures. It is based on the assumption that the right image frame consists of the contents of the horizontal translated version of the left image frame. Motion compensation was also utilized to exploit the temporal redundancy in the right image sequence. An improved disparity estimation was proposed and was demonstrated to be an effective method for the improvement of the coding scheme.

To test the stereo coding scheme, a software prototype encoder was constructed. An objective test based on the peak signal-to-noise ratio was performed to investigate the reproduced right image sequence. It was found that the *PSNR* of the reproduced right image sequence depends on the transmission bit rate as well as the image complexity of the video sequence. It was concluded that higher quality of the right image sequence is reproduced for the same bit rate as the normal MPEG coded right image sequence; hence improvement of the compression ratio is achieved by reducing the bit rate for the transmission of the stereoscopic information to get same image quality. Moverover, less *PSNR* is obtained for image sequences containing stronger depth perception with the same transmission bit rate and therefore the reproduced image quality is depth perception dependent. As the right image sequence is reproduced with reference to the left image sequence, the reconstructed image quality of the left image sequence will also affect that of the right image sequence. In parallel, subjective test was conducted by displaying the reproduced stereo image pairs on screen and comparing them visually. The result showed that visual quality degrade is not observable for less than 1 dB *PSNR* discrepancy

and comparing to coding the two channels of stereo video separately in MPEG, the new stereo coding scheme gave 15% further compression with no substantial visual degrade in image quality.

This scheme not only provides a more efficient way to code stereoscopic motion video but also retains the compatibility with MPEG standard, which is gaining immense population in CD-ROM multimedia applications. Although MPEG aims at data transfer rates around 1.5 Mbits/sec and the inclusion of stereoscopic information requires extra bandwith, double speed or even quad speed CD-ROM drives are becoming standard peripherals in computer systems. Hence, the higher data transfer rate requirement will not post to be a problem.

Appendix A

MPEG — An International Standard

A.1 Introduction

The MPEG [49] is an ISO/IEC (International Organisation for Standardization/International Electrotechnical Commission) standard and is named after the Moving Pictures Expert Group, which is part of the ISO/IEC JTC1/SC29 WG11, started it. More specifically, MPEG does not specify one standard but four standards: MPEG-1 for storage applications at bit rates around 1.5 Mbits/sec, MPEG-2 aiming at higher bit rates for broader applications including telecommunications and broadcasting, MPEG-3 aiming at coded bit rates between 20 and 40 Mbits/sec for high definition TV (HDTV) applications, and MPEG-4 for very low bit rate applications (between 4.8 and 64 kbits/sec) including videophones, multimedia electronic mail, remote sensing and interactive multimedia. Among them MPEG-1 and MPEG-2 have become international standards while the others are still drafting. MPEG-2 is more versatile than MPEG-1. It provides compatibility to the MPEG-1 standard because its syntax is a superset of the MPEG-1 syntax. Many options and extra parameters not supported by MPEG-1 are specified in various extension headers. On considering that the options and extra parameters of MPEG-2 are not used, MPEG-1 video algorithm was selected to be the base algorithm in this research.

In the following discussion, the term "MPEG" represents "MPEG-1" for brevity.

The MPEG specification is defined in three parts: video, audio and system. The MPEG video and MPEG audio specify the compression of video sequence and digital audio signals, respectively, while the MPEG system specifies the encapsulation (or the multiplexing methods) of the data streams generated by MPEG video and MPEG audio. They are optimized to generate a combined bitstream with approximately 1.5 Mbits/sec combined data rates. The MPEG video coding algorithm achieve compression by employing the two-dimensional DCT to reduce spatial redundancy, and motion compensation to reduce temporal redundancy. The MPEG coding algorithm was developed primarily for storage of compressed video on digital storage media including conventional storage devices such as CD-ROM, DAT, tape drives, hard disks, and writable optical drives. Like a normal video tape recording system, it must support features such as random access and fast forward/reverse searches. In this appendix the outline of the coding algorithm with emphasis on how the basic theories discussed in the chapter 2 can be applied and the coding of important quantities are presented.

A.2 Preprocessing

It is described in the chapter 2 that motion video can be considered as a set of image frames displayed sequentially at the rate 30 frames/sec. A digital image frame is usually expressed as two dimensional array of pixels, which is denoted by three color values consisting of red, green and blue. The three color values representing a pixel can be considered as a *RGB triplets*. For the analog video source, sampling is required before MPEG coding.

As MPEG algorithm operates on the Y-Cr-Cb color space, where Y and Cr/Cb represent the luminance level and the two orthogonal chrominance levels respectively, each RGB triplet has to be converted into a YCrCb triplet. Due to the fact that human visual

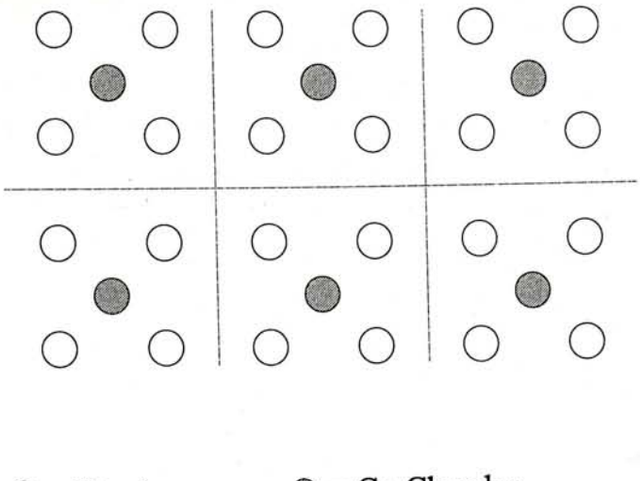


Figure A.1: *Luminance and chrominances of pixels.*

system is more sensitive to luminance or intensity than chrominances of a pixel, the Cr and Cb values are subsampled so that every 4×4 pixels correspond to four Y values, one Cr value and one Cb value. It is shown in Figure A.1, where the location of the Cb and Cr values is the same; thereby only one circle is drawn in the figure.

A.3 Data Structure of Pictures

The MPEG video bitstream utilizes layered approach. The top video level of coding is called sequence layer, which consists of a number of groups of pictures (GOPs), where the first picture of a GOP is always an intra-coded picture and the rest of pictures may be some number of predicted pictures and bidirectionally predicted pictures. The three different picture types are discussed in the next section. The coding syntax of the pictures is organized in four coding layers: block layer, macroblock layer, slice layer and picture layer. Each coding layer encodes the corresponding elements, i.e., block, macroblock, slice and picture.

A block is the coding syntax that describes the smallest unit in MPEG video terminology. It is actually a 8×8 DCT coefficients unit corresponding to 8×8 values of one of the

three types: luminance (Y), red chrominance (Cr), and blue chrominance (Cb).

A 16×16 pixel segment converted into four 8×8 Y blocks and two 8×8 chrominance blocks, one for Cr and the other for Cb, forms a macroblock which is the basic coding unit in MPEG video terminology. There are two coding modes available at the macroblock layer. They are the intracoded and motion compensated modes. As implied by their names, intracoded mode indicates that the six 8×8 blocks are coded by intra processing only. The DCT coefficients contained in the blocks are directly transformed from the 16×16 pixel segment in the image frame. For motion compensated coding mode, the DCT coefficients contained in the six blocks are transformed from the differences between the 16×16 pixel segment in the current and the previous image frames.

A slice is composed of one or more contiguous macroblocks. Because of the independent coding attribute, a slice containing errors can be skipped and the next slice is not affected by those errors. It is obvious that slices facilitate the error concealment. However, having more slices requires more bits for overheads of slice layers. The MPEG video standard does not define its length. For most applications, it is defined to be a horizontal strip within an image frame.

A picture defined by MPEG is the basic unit for display. It corresponds to an image frame. In this thesis, the term "image frame" denotes a bitmapped digital image which is a two dimensional array of pixels while the term "picture" denotes the MPEG coded representation of a digital image. Figure A.2 depicts the MPEG data structure schematically.

A.4 Picture Coding

MPEG defines three different picture types:

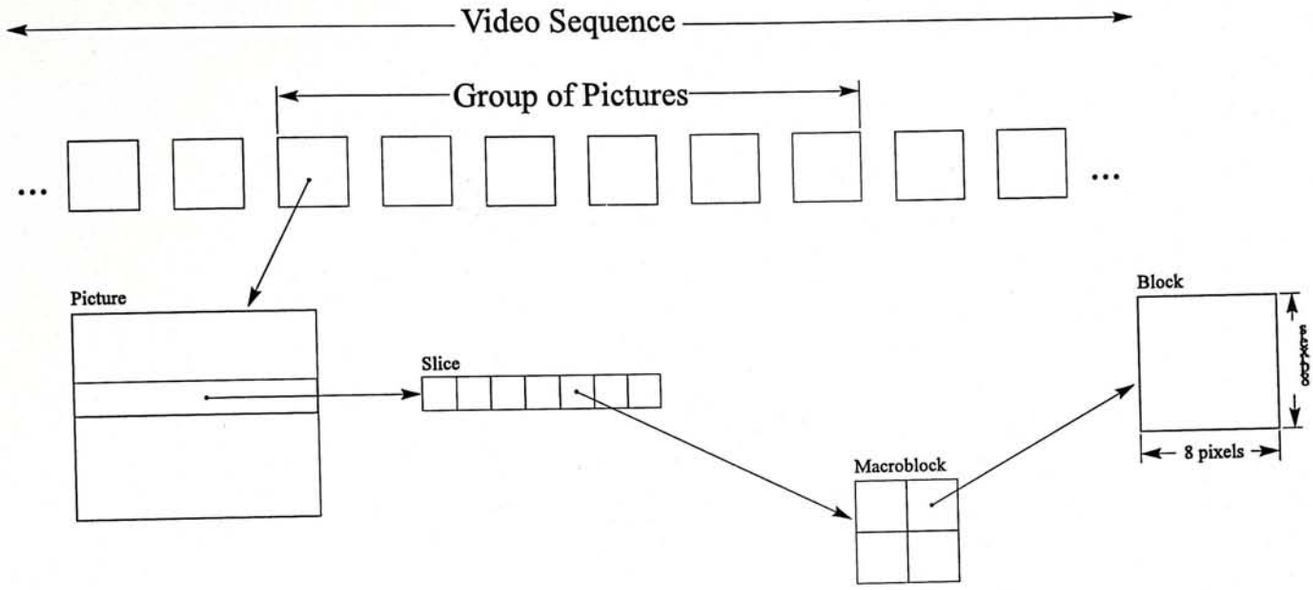


Figure A.2: *Schematic diagram of MPEG data structure.*

- *Intracoded pictures* (I-pictures) are image frames coded by intraframe processing only. Each image frame to be coded into I-picture is first divided into 16×16 pixel segments. Each segment is then converted into the corresponding macroblock. Image frames coded into I-pictures can only contain macroblocks coded in intracoded mode, which is generated by applying a two dimensional 8×8 DCT to all luminance and chrominance blocks. Like the general transform coding operation described in the last chapter, a quantizer is utilized to quantize the transform coefficients. It is a scalar quantizer with different step size for the 64 DCT coefficients. The step size is obtained from an 8×8 frequency dependent quantization matrix, which can be either the default quantization matrix derived by the MPEG video committee or determined by the encoder and then transmitted to the decoder. The use of the matrix ensures that the low frequency DCT coefficients are quantized more accurately with a small step size while the high frequency coefficients are quantized more coarsely. This reflects the human visual system which is less sensitive to quantization noise at higher frequencies.

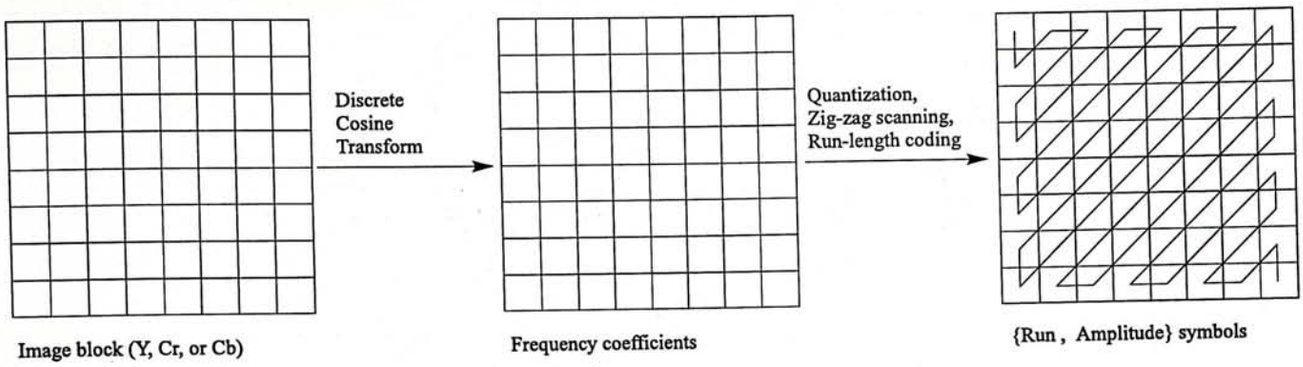


Figure A.3: *The coding of intra-coded pictures.*

After the quantization, a significant proportion of the quantized coefficients becomes zero. These coefficients are then scanned in a zigzag manner to create a sequence of quantized values with the longest run of zeros. It is beneficial to apply run-length and truncated Huffman coding by grouping each pair of zero-run and the following non-zero quantized values into an "event" or a {run, amplitude} pair. For the most frequent combinations of {run, amplitude} pairs MPEG defines Huffman coding table while for the {run, amplitude} pairs not defined a fixed length code is used instead. This coding strategy is the truncated Huffman coding scheme described previously. Figure A.3 depicts the coding of I-pictures. Being coded without reference to other pictures, I-pictures provide random access points.

- *Predicted pictures* (P-pictures) are image frames coded using motion-compensated predictions with reference to the previous I- or P-picture. Like I-pictures, the coding of P-pictures is also processed on a macroblock basis. Therefore, the image frame to be coded is divided into 16×16 pixel segments. Before converted into the corresponding macroblock, each segment undergoes motion estimation, in which motion vector minimizing the absolute difference between the current pixel segment and a displaced pixel segment in the previous image frame is identified by searching a square area around each segment. Since the previous image frame is required for the motion compensation process, the previous I- or P-picture must be decoded into image frame before the coding of the current P-picture. Thus, there

is a decoder embeded in an encoder.

The macroblocks contained in a P-picture can be intracoded and motion compensated modes. The determination of the modes depends on the prediction error between the current pixel segment and the motion compensated pixel segment. If this prediction error is smaller than a threshold or the total energy in the segment, the pixel segment is coded as a motion compensated macroblock; therefore, the motion vector corresponding to this pixel segment is variable length coded and transmitted. In addition, the prediction error, after motion compensation is applied to the pixel segment, is encoded using DCT. The transform coefficients of the prediction error are then quantized. Unlike the intraframe mode, the quantization step sizes in this case are constant for all 64 coefficients. The quantized coefficients are then variable length coded as in the intraframe mode. If, however, the prediction error is larger than the total energy in the current segment, motion compensated mode is not suitable and all the six blocks in the corresponding macroblock are coded using the intracoded mode as described for the coding of I-pictures. Since temporal redundancy is reduced by motion compensation, P-pictures provide more compression than those provided by I-pictures.

- *Bidirectionally predicted pictures* (B-pictures) are image frames coded using both a past and future pictures, which can be an I- and/or P-picture, as references to apply interpolation. The coding of B-pictures is very similar to that of P-pictures except that both forward, backward and interpolative motion compensation processes are used for the macroblocks coded in the motion compensated mode. Requiring the future image frame as reference, the coding of B-pictures is a non-causal operation and, therefore, the coded future image frame either an I- or P-picture is coded and then decoded back into image frame before the coding of the current B-picture. Two image frame buffers are required one for the past image frame and the other for the future image frame. Forward prediction with reference to a past image frame, backward prediction with reference to a future image frame and interpolation by

Macroblock Type	Predictor	Prediction Error
Intra	$\hat{I}_1(\bar{X}) = 128$	$I_1(\bar{X}) - \hat{I}_1(\bar{X})$
Forward Predicted	$\hat{I}_1(\bar{X}) = \hat{I}_0(\bar{X} + \overline{\mathrm{MV}_{01}})$	$I_1(\bar{X}) - \hat{I}_1(\bar{X})$
Backward Predicted	$\hat{I}_1(\bar{X}) = \hat{I}_2(\bar{X} + \overline{\mathrm{MV}_{21}})$	$I_1(\bar{X}) - \hat{I}_1(\bar{X})$
Interpolative	$\hat{I}_1(\bar{X}) = \frac{1}{2}\left[\hat{I}_0(\bar{X} + \overline{\mathrm{MV}_{01}}) + \hat{I}_2(\bar{X} + \overline{\mathrm{MV}_{21}})\right]$	$I_1(\bar{X}) - \hat{I}_1(\bar{X})$

where $\hat{I}_n$ represents the predicted value of I_n; the suffix n indicates the order of image frame I_n and $\overline{\mathrm{MV}_{nm}}$ represents the motion vector of the macroblock for image frame changing from I_n to I_m.

Table A.1: Prediction Modes for Macroblock in B-picture

averaging both the forward prediction and the backward prediction pixel segment of the 16×16 pixel segment is obtained. The one that gives the smallest prediction error is selected. This prediction error is also compared with the threshold. If the prediction error is larger than the threshold, intracoded mode is applied to code the corresponding macroblock; otherwise, the motion vectors for the selected type (i.e., forward motion vector for forward prediction, backward motion vector for backward prediction, and both motion vectors for interpolation) are variable length coded and transmitted with the DCT encoded prediction error. Table A.1 describes the prediction modes for macroblocks in B-picture by mathematical symbols. It is clear that B-pictures although provide the most compression, they require more compution as well as extra memory.

The flow chart for coding a macroblock is depicted in Figure A.4.

A.4.1 Coding of Motion Vectors

As the motion of objects in an image sequence is correlated, motion vectors are coded by differential means to make use of this feature. Motion vectors can be in either full-pixel or half-pixel units depending on the setting of the bit flags, full_pel_forward_vector and full_pel_backward_vector, in the picture header. The range of the vectors is determined not only by the full_pel flag but also forward_f_code or backward_f_code, which is an

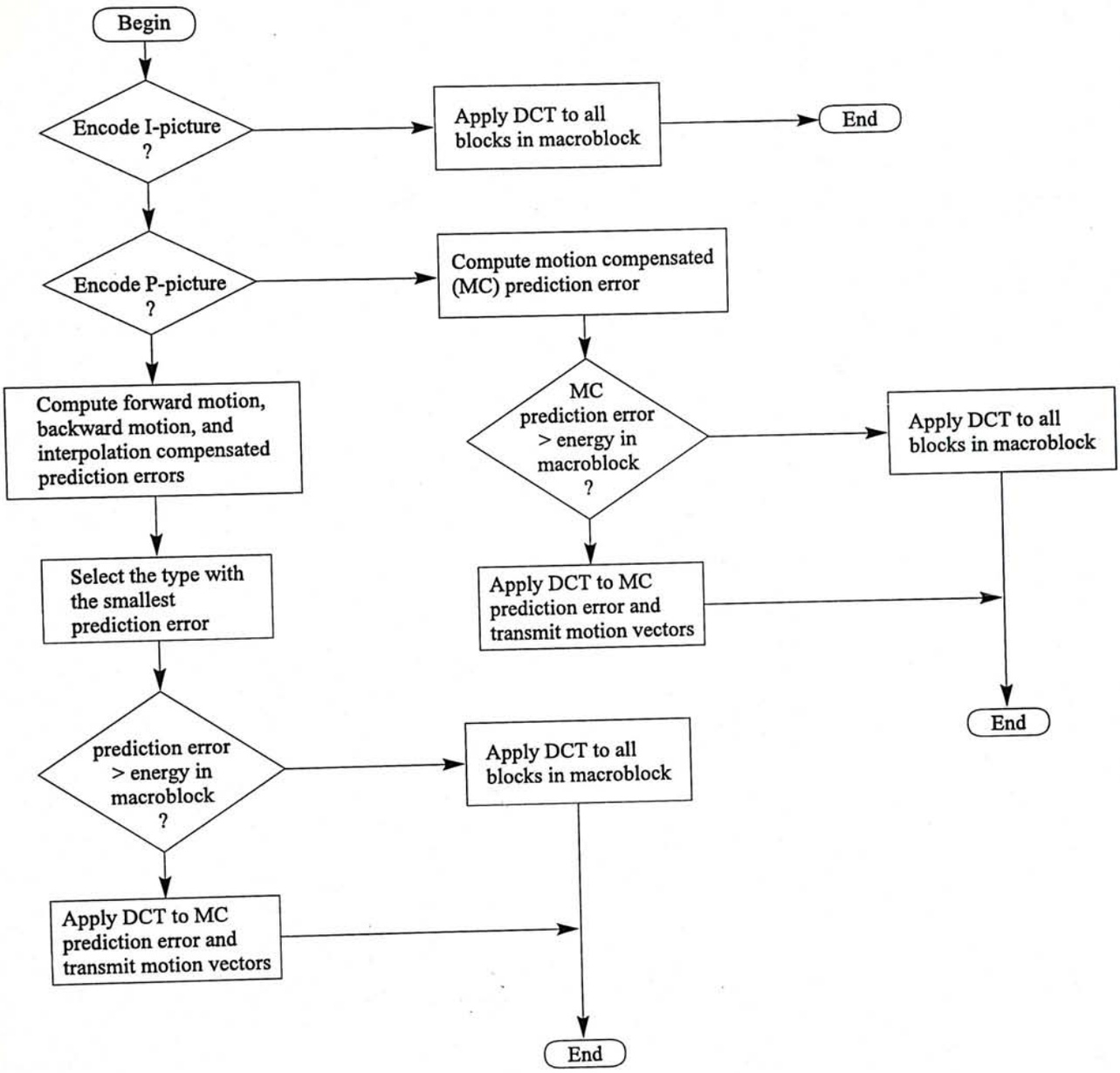


Figure A.4: *The flow chart for macroblock coding.*

forward_f_code or backward_f_code	Motion vector range	
	full_pel = 0	full_pel = 1
1	-8 to 7.5	-16 to 15
2	-16 to 15.5	-32 to 31
3	-32 to 31.5	-64 to 63
4	-64 to 63.5	-128 to 127
5	-128 to 127.5	-256 to 255
6	-256 to 255.5	-512 to 511
7	-512 to 511.5	-1024 to 1023

Table A.2: Range of Motion Vectors

unsigned 3-bit integer in the picture header. Table A.2 shows the range of motion vectors. From the forward_f_code, two important parameters for coding the forward motion vectors, forward_r_size and forward_f, are calculated by the following equations.

$$\begin{aligned} \text{forward\_r\_size} &= \text{forward\_f\_code} - 1 \\ \text{forward\_f} &= 1 \ll \text{forward\_r\_size} \end{aligned} \tag{A.1}$$

From the backward_f_code, the two parameters for backward motion vectors are calculated as follows:

$$\begin{aligned} \text{backward\_r\_size} &= \text{backward\_f\_code} - 1 \\ \text{backward\_f} &= 1 \ll \text{backward\_r\_size} \end{aligned} \tag{A.2}$$

The variable length coding of motion vectors consists of a constrained set of differential motion code as shown in Table A.3 followed by unsigned integer code with the fixed bit length, forward_r_size, to represent the unsigned remainder which is obtained by adding or subtracting the modulus given in Table A.4 according to the f_code. The f_code either forward or backward is determined by considering motion vectors of macroblocks in a slice with the most suitable range. For example a slice consisting of eight macroblocks has the following motion vectors with the full_pel flag setting to "1".

7 3 -1 -20 55 25 -14 -16

Variable Length Code	Differential value
0000 0011 001	-16
0000 0011 011	-15
0000 0011 101	-14
0000 0011 111	-13
0000 0100 001	-12
0000 0100 011	-11
0000 0100 11	-10
0000 0101 01	-9
0000 0101 11	-8
0000 0111	-7
0000 1001	-6
0000 1011	-5
0000 11	-4
0001 1	-3
0011	-2
011	-1
1	0
010	1
0010	2
0001 0	3
0000 110	4
0000 1010	5
0000 1000	6
0000 0110	7
0000 0101 10	8
0000 0101 00	9
0000 0100 10	10
0000 0100 010	11
0000 0100 000	12
0000 0011 110	13
0000 0011 100	14
0000 0011 010	15
0000 0011 000	16

Table A.3: Differential Motion Code

forward_f_code or backward_f_code	Modulus
1	32
2	64
3	128
4	256
5	512
6	1024
7	2048

Table A.4: Modulus for Motion Vectors

The most suitable range selected is -64 to 63 which corresponds to f_code of 3. The differential values are given by setting the initial prediction in each slice to zero and are shown as follows:

7 -4 -4 -19 75 -30 -39 -30

By adding or subtracting the modulus 128 corresponding to the f_code of 3 reduces the differential values to the range -64 to 63:

7 -4 -4 -19 -53 -30 -39 -30

The above values are then divided by either forward_f or backward_f, which is give by the equation (A.1) or (A.2) and is 4 in this example. Therefore, the following codes expressed as (differential value, remainder) are obtained:

(1,3) (-1,0) (-1,0) (-5,1) (-14,3) (-8,2) (-10,1) (-8,2)

The corresponding codes for the motion vectors shown as follows are obtained by getting the differential motion codes from Table A.3 followed by the unsigned remainders with the fixed bit length, forward_r_size, which is 2 in this example.

VLC of motion vectors in binary	(differential value, remainder)
011	(1,3)
0110 0	(-1,0)
0110 0	(-1,0)
0000 1011 01	(-5,1)
0000 0011 1011 1	(-14,3)
0000 0101 1110	(-8,2)
0000 0100 1101	(-10,1)
0000 0101 1110	(-8,2)

A.4.2 Coding of Quantized Coefficients

The top left coefficient in Figure A.3 is the DC coefficient while the rest of the coefficients is described as AC. The DC coefficient depicts the average luminance or chrominance of

the image block; thus it tends to be well correlated with the DC coefficient of the preceding block. DC coefficients are coded by differential means to make use of this feature. However, the AC coefficients are not correlated and, therefore, are coded independently.

DC Coefficients

The differential DC values after quantization are categorized according to their absolute value as shown in Table A.5. The size specifying the number of additional bits required to define the amplitude of the DC coefficient is transmitted using a VLC code. The different code for luminance and chrominance is because their statistics are different. If the DC coefficient in the category '32 to 63', a size of 6 is transmitted by either '1111 0' for luminance or '1111 10' for chrominance followed by six additional bits to represent the amplitude with the first of these additional bits indicating the sign: '0' for negative and '1' for positive. In addition, if a size of 0 is transmitted, there is no additional bits. Table A.6 shows the corresponding additional bits. Suppose the change of the DC coefficient for luminance is 25, which belongs to the range '16 to 31'. From Table A.5, the size to be transmitted is 5 with the VLC code '1110' followed by five additional bits. Since the change is positive, the first bit of the additional bits is '1'. The value 25 is than directly convert to binary with '11001'. Thus, the code for the change of 25 in luminance is '111011001'.

AC Coefficients

The AC quantized coefficients are coded as {run, amplitude} pairs after scanned in the zigzag manner as shown in Figure A.3 previously. For the sake of clarity example is used

Differental DC (absolute value)	size	VLC code (luminance)	VLC code (chrominance)
0	0	100	00
1	1	00	01
2 to 3	2	01	10
4 to 7	3	101	110
8 to 15	4	110	1110
16 to 31	5	1110	1111 0
32 to 63	6	1111 0	1111 10
64 to 127	7	1111 10	1111 110
128 to 255	8	1111 110	1111 1110

Table A.5: Differential DC size and VLC.

Differental DC	size	additional code
-255 to -128	8	00000000 to 01111111
-127 to -64	7	0000000 to 0111111
-63 to -32	6	000000 to 011111
-31 to -16	5	00000 to 01111
-15 to -8	4	0000 to 0111
-7 to -4	3	000 to 011
-3 to -2	2	00 to 01
-1	1	0
0	0	
1	1	1
2 to 3	2	10 to 11
4 to 7	3	100 to 111
8 to 15	4	1000 to 1111
16 to 31	5	10000 to 11111
32 to 63	6	100000 to 111111
64 to 127	7	1000000 to 1111111
128 to 255	8	10000000 to 11111111

Table A.6: Differential DC additional code.

to depict the coding process. Suppose the block of quantized coefficients is as follows:

27	0	0	0	0	0	0	0
10	−6	−1	0	0	0	0	0
7	0	2	0	0	0	0	0
2	0	0	120	0	0	0	0
0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0

After zigzag scanned, the {run, amplitude} pairs are shown as follows. The first coefficient, 27, is the DC coefficient, which is coded separately as described previously and is ignored in AC coefficients coding.

run-length	amplitude
1	10
0	7
0	-6
2	-1
1	2
2	2
11	120
end	

The {run, amplitude} pairs are then coded using Table A.7. In the table, the last bit 's' denotes the sign of the amplitude. For s = 0, the amplitude is positive; while for s = 1, the amplitude is negative. The symbol 'EOB' means end_of_block, which is transmitted indicating the end of the transmission of the current block of DCT coefficients. The table only provides the most possible combinations of {run, amplitude}. For those pairs not given in Table A.7, the escape code is used followed by the VLC code shown in Table A.8 to represent the run-length and the VLC code shown in Table A.9 to represent the

run	amplitude	VLC code
EOB		10
0	1	1s for first coeff
0	1	11s for next coeff
0	2	0100 s
0	3	0010 1s
0	4	0000 110s
0	5	0010 0110 s
0	6	0010 0001 s
0	7	0000 0010 10s
0	8	0000 0001 1101 s
0	9	0000 0001 1000 s
0	10	0000 0001 0011 s
0	11	0000 0001 0000 s
0	12	0000 0000 1101 0s
0	13	0000 0000 1100 1s
0	14	0000 0000 1100 0s
0	15	0000 0000 1011 1s
0	16	0000 0000 0111 11s
0	17	0000 0000 0111 10s
0	18	0000 0000 0111 01s
0	19	0000 0000 0111 00s
0	20	0000 0000 0110 11s
0	21	0000 0000 0110 10s
0	22	0000 0000 0110 01s
0	23	0000 0000 0110 00s
0	24	0000 0000 0101 11s
0	25	0000 0000 0101 10s
0	26	0000 0000 0101 01s
0	27	0000 0000 0101 00s
0	28	0000 0000 0100 11s
0	29	0000 0000 0100 10s
0	30	0000 0000 0100 01s
0	31	0000 0000 0100 00s
0	32	0000 0000 0011 000s
0	33	0000 0000 0010 111s
0	34	0000 0000 0010 110s
0	35	0000 0000 0010 101s
0	36	0000 0000 0010 100s
0	37	0000 0000 0010 011s
0	38	0000 0000 0010 010s
0	39	0000 0000 0010 001s
0	0	0000 0000 0010 000s
1	1	011s
1	2	0001 10s
1	3	0010 0101 s
1	4	0000 0011 00s
1	5	0000 0001 1011 s
1	6	0000 0000 1011 0s
1	7	0000 0000 1010 1s
1	8	0000 0000 0011 111s
1	9	0000 0000 0011 110s
1	10	0000 0000 0011 101s
1	11	0000 0000 0011 100s
1	12	0000 0000 0011 011s
1	13	0000 0000 0011 010s
1	14	0000 0000 0011 001s
1	15	0000 0000 0001 0011

run	amplitude	VLC code
1	16	0000 0000 0001 0010 s
1	17	0000 0000 0001 0001 s
1	18	0000 0000 0001 0000 s
2	1	0101 s
2	2	0000 100s
2	3	0000 0010 11s
2	4	0000 0001 0100 s
2	5	0000 0000 1010 0s
3	1	0011 1s
3	2	0010 0100 s
3	3	0000 0001 1100 s
3	4	0000 0000 1001 1s
4	1	0011 0s
4	2	0000 0011 11s
4	3	0000 0001 0010 s
5	1	0001 11s
5	2	0000 0010 01s
5	3	0000 0000 1001 0s
6	1	0001 01s
6	2	0000 0001 1110 s
6	3	0000 0000 0001 0100 s
7	1	0001 00s
7	2	0000 0001 0101 s
8	1	0000 111s
8	2	0000 0001 0001 s
9	1	0000 101s
9	2	0000 0000 1000 1s
10	1	0010 0111 s
10	2	0000 0000 1000 0s
11	1	0010 0011 s
11	2	0000 0000 0001 1010 s
12	1	0010 0010 s
12	2	0000 0000 0001 1001 s
13	1	0010 0000 s
13	2	0000 0000 0001 1000 s
14	1	0000 0011 10s
14	2	0000 0000 0001 0111 s
15	1	0000 0011 01s
15	2	0000 0000 0001 0110 s
16	1	0000 0010 00s
16	2	0000 0000 0001 0101 s
17	1	0000 0001 1111 s
18	1	0000 0001 1010 s
19	1	0000 0001 1001 s
20	1	0000 0001 0111 s
21	1	0000 0001 0110 s
22	1	0000 0000 1111 1s
23	1	0000 0000 1111 0s
24	1	0000 0000 1110 1s
25	1	0000 0000 1110 0s
26	1	0000 0000 1101 1s
27	1	0000 0000 0001 1111 s
28	1	0000 0000 0001 1110 s
29	1	0000 0000 0001 1101 s
30	1	0000 0000 0001 1100 s
31	1	0000 0000 0001 1011 s
ESCAPE	-	0000 01

Table A.7: Combination Codes for DCT quantized coefficients.

run-length	VLC code
0	0000 00
1	0000 01
2	0000 10
⋮	⋮
62	1111 10
63	1111 11

Table A.8: Zero run-length codes

amplitude	VLC code
-256	forbidden
-255	1000 0000 0000 0001
-254	1000 0000 0000 0010
⋮	⋮
-129	1000 0000 0111 1111
-128	1000 0000 1000 0000
-127	1000 0001
-126	1000 0010
⋮	⋮
-2	1111 1110
-1	1111 1111
0	forbidden
1	0000 0001
2	0000 0010
⋮	⋮
126	0111 1110
127	0111 1111
128	0000 0000 1000 0000
129	0000 0000 1000 0001
⋮	⋮
254	0000 0000 1111 1110
255	0000 0000 1111 1111

Table A.9: Amplitude codes for DCT AC quantized coefficients.

amplitude. According to Tables A.7 to A.9, the VLC codes for the {run, amplitude} pairs in the example above are obtained as follows:

run-length	amplitude	code	description
1	10	0000 0000 0011 1010	s=0 for positive
0	7	0000 0010 100	s=0 for positive
0	-6	0010 0001 1	s=1 for negative
2	-1	0101 1	s=1 for negative
1	2	0001 100	s=0 for positive
2	2	0000 1000	s=0 for positive
11	120	0000 0100 1011 0111 1000	escape code + run + amplitude
EOB		10	end of block

After the transmission of the non-zero coefficient, the EOB code is transmitted to inform the decoder that there is no more coefficient in the current 8×8 block.

References

[1] D. A. Huffman. "A method for the construction of minimum redundancy codes." In *Proceedings of the IRE*, volume 40, pages 1098–1101, 1952.

[2] J. Ziv and A. Lempel. "Compression of individual sequences via variable-rate coding." *IEEE Transactions on Information Theory*, IT-24:pp530–536, 1978.

[3] C. E. Shannon. "A mathematical theory of communication." *The Bell System Technical Journal*, XXVII(3):pp379–423, 1948.

[4] M. Vetterli. "Filter banks allowing perfect reconstruction." *Signal Processing*, 10:pp219–244, April 1986.

[5] J. W. Wood and S. D. O'Neill. "Subband coding of images." *IEEE Transactions on Acoustics, Speech, and Signal Processing*, ASSP-43:pp1278–1288, October 1986.

[6] T. Koga *et al.*. "Motion compensated interframe coding for video conferencing." *National Telecommun. Conf.*, NTC-81:G5.3.1–G5.3.5, Nov.-Dec. 1981.

[7] J. R. Jain and A. K. Jain. "Displacement measurement and its application in interframe image coding." *IEEE Trans. Communication*, COM-29:pp1799–1808, December 1981.

[8] R. Srinivasan and K. R. Rao. "Predictive coding based on efficient motion estimation." *Intl. Conf. Commun.*, ICC-88:pp521–526, May 1984.

[9] Bjørn Olstad. "Adaptive temporal decimation for video compression algorithms." *Journal of Electronic Imaging*, 2:pp5–18, January 1993.

[10] J. A. Roese, W. K. Pratt, and G. S. Robinson. "Interframe cosine transform image coding." *IEEE Trans. Commun.*, COM-25:pp1329–1338, Nov. 1977.

[11] T. R. Natarajan and N. Ahmed. "On interframe transform coding." *IEEE Trans. Commun.*, COM-25:pp1323–1329, Nov. 1977.

[12] J. K. Aggarwal and N. Nandhakumar. "On the computation of motion from sequences of images – a review." *Proceedings of the IEEE*, 76(8):pp917–935, 1988.

[13] G. Adiv. "Determining three-dimensional motion and structure from optical flow generated by several moving objects." *IEEE Trans. Pattern Analysis and Machine Intelligence*, 7(4):pp384–401, 1985.

[14] R. Thoma and M. Bierling. "Motion compensating interpolation considering covered and uncovered background." *Image communication*, 1(2):pp191–212, 1989.

[15] J. S. Lim. "Two-Dimensional Signal and Image Processing." Prentice Hall Inc., Englewood Cliffs, N.J., 1990.

[16] H. Karhunen. "Uber Lineare Methoden in der Wahrsheinlich-Keitsrechnung." *Ann. Acad. Science Fenn*, Ser. A.I. 37, Helsmki, 1947.

[17] M. Loève. "Fonctions Aleatoires de Seconde Ordre." *Processus Stochastiques et Mouvement Brownien*, P. Levy, 1948.

[18] H. Hotelling. "Analysis of a Complex of Statistical Variables into Principle Components." *J. Educ. Psychology*, 24:pp417–441 and pp498–520, 1933.

[19] W. D. Ray and R. M. Driver. "Further decomposition of the Karhunen-Loève series represetntation of a stationary random process." *IEEE Transactions on Information Theory*, IT-16:pp663–668, Nov. 1970.

[20] N. Ahmed, T. Natarajan, and K. R. Rao. "Discrete cosine transforms." *IEEE Transactions on Computer*, C-23:pp90–93, 1974.

[21] R. J. Clark. "Transform Coding of Images". Academic Press, New York, 1985.

[22] H. Malvar and D. Staelin. "The LOT: Transform coding without blocking effects." *IEEE Transactions on Acoustics, Speech, and Signal Processing*, 37:pp553–559, April 1989.

[23] S. G. Mallat. "A theory for multiresolution signal decomposition: The wavelet representation." *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 11(7):pp674–693, 1987.

[24] H. Gharavi and A. Tabaatabai. "Application of quadrature mirror filters to the coding of monochrome and color images." *Proceedings ICASSP*, pages 1–32,8,4, 1987.

[25] A. Tran, K. Liu, K. Tzou, and E. Vogel. "An efficient pyramid image coding system." *Proceedings ICASSP*, pages 18.6.1–18.6.4, 1987.

[26] I. Daubechies. Ten lectures on wavelets. "Society for Industrial and Applied Mathematics", 1992. Philadelphia, PA.

[27] J. Max. "Quantizing for minimum distortion." *IRE Transactions on Information Theory*, IT-6:pp7–12, 1960.

[28] J. B. O'Neil. "Entropy coding in speech and television differential pcm systems." *IEEE Transactions on Information Theory*, IT-17:pp758–761, 1971.

[29] Y. Linde, A. Buzo, and R. M. Gray. "An algorithm for vector quantizer design." *IEEE Transactions on Communication*, COM-28:pp84–95, January 1980.

[30] R. M. Gray. "Vector quantization." *ASSP Magazine*, 1:pp4–29, April 1984.

[31] D. G. Jeong and J. D. Gibson. "Lattice vector quantization for image coding." *Proceedings ICASSP*, pages 1743–1746, 1989.

[32] A. K. Jain. "Fundamentals of Digital Image Processing." Prentice Hall Inc., Englewood Cliffs, N.J., 1989.

[33] A. W. Davis. "Desktop Videoconferencing and Imageing: Is there really an H.320 vs. Indeo Conferencing compression war?" *Advanced Imaging*, September 1994.

[34] CCITT SG XV. "Draft Revision of Recommendation H.261: Video Codec for Audiovisual Services at $p \times 64$ kbit/sec." *Signal Processing: Image Communication*, 2(2), August 1990.

[35] P. Hilaire, S. Benton, and *et al.*. "Electronic display system for computational holography." *SPIE*, 1212-20, 1990.

[36] Rüdiger Sand. "3-DTV Research and Development in Europe." *ITEC'91 : 1991 ITE Annual Convention*, 1991.

[37] I. Sutherland. "A head-mounted three-dimensional display." *FJCC*, 33:pp757–764, 1968.

[38] H. Isono and M. Yasuda. "Autostereoscopic 3D-TV display research at NHK." *ITEC'91 : 1991 ITE Annual Convention*, 1991.

[39] L. Lipton. "True Stereoscopic Television: 3DTV is feasible." *Advanced Imaging*, September 1994.

[40] K. Gomi, Y. Nishino, and K. Tai. "Stereoscopic video transmission and presentation system for ISDN." *IEEE Transaction on Consumer Electronics*, 36(3), August 1990.

[41] H. Yamaguchi, Y. Tatehira, K. Akiyama, and Y. Kobayashi. "Statistical characteristics of stereoscopic images for image coding." In *Three-Dimensional Visualization and Display Technologies*, volume 1083, pages 135–142. SPIE, 1989.

[42] M. Ziegler and R. Sand. "Stereoscopic imaging." In *ISO/MPEG- ISO/IEC JTC1/SC29/WG11, Doc. MPEG 92/328*, Rio de Janeiro, BRASIL, 1992.

[43] I. Dinstein, G. Guy, J. Rabany, J. Tzelgov, and A. Henik. "On stereo image coding." In *Proc. 9th International Conference on Pattern Recognition*, pages 357–359, Rome, Italy, November 1988.

[44] M. E. Lukacs. "Predictive coding of multi-viewpoint image sets." *Proceedings ICASSP*, pages 521–524, 1986.

[45] S. K. Ip and S. M. Chiang. "Stereoscopic Video Coding for the Applications in Virtual Reality." *Third International Symposium on Consumer Electronics, IEE ISCE'94*, pages 241 – 245, November 1994.

[46] V. Seferidis and M. Chanbari. "Generalized block matching motion estimation." In *Visual Communications and Image Processing*, volume 1818, pages 110–119. SPIE, 1992.

[47] W. H. Chen, C. H. Smith, and S. C. Fralick. "A fast computational algorithm for the discrete cosine transform." *IEEE Trans. Communication*, COM-25:pp1004–1009, 1977.

[48] ISO-IEC JTC1/SC2/WG8. "MPEG Video Simulation Test Model Three SM3", 1990.

[49] Moving Picture Experts Group. "ISO CD11172-2: Coding of moving pictures and associated audio for digital storage media at up to about 1.5 Mbit/s", November 1991.